

A Network Coding Approach to Loss Tomography

Athina Markopoulou,^{†*} Christina Fragouli,[‡] Pegah Sattari,[†] Minas Gjoka[†]

[†]University of California, Irvine and [‡]EPFL, Switzerland

^{*}(Please send correspondence to: *email*: athina@uci.edu, *tel*: +1 (949) 824-0357, *fax*: +1 (949) 824-8197, *mail*: CalIT2 Bldg 4100, Irvine, CA 92697-2800, USA)

Abstract

Network tomography aims at inferring internal network characteristics based on measurements at the edge of the network. In loss tomography, in particular, the characteristic of interest is the loss rate of individual links. There is a significant body of work dedicated to this problem using multicast and/or unicast end-to-end probes. Independently, recent advances in network coding have shown that there are several advantages from allowing intermediate nodes to process and combine, in addition to just forward, packets. In this paper, we pose the problem of loss tomography in networks that have network coding capabilities. We design a framework for estimating link loss rates, which leverages network coding capabilities and we show that it improves several aspects of tomography, including the identifiability of links, the tradeoff between estimation accuracy and bandwidth efficiency, and the complexity of probe path selection. We discuss the cases of inferring the loss rates of links on a tree topology or on a general topology. In the latter case, the benefits of our approach are even more pronounced compared to standard techniques but we also face novel challenges, such as dealing with cycles and multiple paths between sources and receivers. Overall, this work was the first to make the connection between tomography and network coding and thus opened a new research direction.

Index Terms

Network Coding, Network Tomography, Link Loss Inference.

I. INTRODUCTION

Distributed Internet applications often need to know information about the characteristics of the network. For example, an overlay or peer-to-peer network may want to detect and recover from failures or degraded performance of the underlying Internet infrastructure. A company with several geographically distributed campuses may want to know the behavior of one or several Internet service providers (ISPs) connecting the campuses, in order to optimize traffic engineering decisions and achieve the best end-to-end performance. To achieve this high-level goal, it is necessary for the nodes participating in the application or overlay to monitor Internet paths, assess and predict their behavior, and eventually make efficient use of them by taking appropriate control and traffic engineering decisions both at the network and at the application layers. Therefore, accurate monitoring at minimum overhead and complexity is of crucial importance in order to provide the input needed to take such informed decisions. However, there is currently no incentive for ISPs to provide detailed information about their internal operation and performance or to collaborate with other ISPs for this purpose. As a result, distributed applications usually rely on their own end-to-end measurements between nodes they have control over, in order to infer performance characteristics of the network.

Over the past decade, a significant research effort has been devoted to a class of monitoring problems that aim at inferring internal network characteristics using measurements at the edge [1]. This class of problems is commonly referred to as *tomography* due to its analogy to medical tomography. In this work, we are particularly interested in loss tomography, *i.e.*, at inferring the loss probabilities (or loss rates) of individual links using active end-to-end measurements [2]–[6]. The topology is assumed known and sequences of probes are sent and collected between a set of sources and a set of receivers at the network edge. Link-level parameters, in this case loss rates of links, are then inferred by the observations at the receivers. The bandwidth efficiency of these methods can be measured by the number of probes needed to estimate the loss rates of interest within a desired accuracy. Despite its significance and the research effort invested, loss tomography remains a hard problem for a number of reasons, including complexity (of optimal probe routing and of estimation), bandwidth overhead, and identifiability (the fundamental fact that tomography is an inverse problem and we cannot directly observe the parameters of interest). Furthermore, there are some practical limitations such as the lack of cooperation of ISPs, the need for synchronization of sources in some schemes, etc.

Recently, a new paradigm to routing information has emerged with the advent of network coding [7]–[9]. The main idea in network coding is that, if we allow intermediate nodes to not only forward but also combine packets, we can obtain significant benefits in terms of throughput, delay and robustness of distributed algorithms. Our work is based on the observation that, in networks equipped with network coding capabilities, we can leverage these capabilities to significantly improve several aspects of loss tomography. For example, with network coding, we can combine probes from different paths into one, thus reducing the bandwidth needed to cover a general graph and also increasing the information per packet. Furthermore, the problem of optimal probe routing, which is known to be NP-hard, can be solved with linear complexity when network coding is used.

This paper proposes a framework for loss tomography (including mechanisms for probe routing, probe and code design, estimation, and identifiability guarantees) in networks that already have network coding capabilities. Such capabilities do not exist yet on the Internet today, but are available in wireless mesh networks, peer-to-peer and overlay networks and we expect them to appear in more environments as network coding becomes more widely adopted. We show that, in those settings, our network coding-based approach improves the following aspects of the loss tomography problem: how many links of the network we can infer (identifiability); the tradeoff between how well we can infer link loss rates (estimation accuracy) and how many probes we need in order to do so (bandwidth efficiency); how to select sources and receivers and how to route probes between them (optimal probe routing). Overall, this is a novel application of network coding techniques to a practical networking problem, and it opens a promising research direction.

The structure of the paper is as follows. Section II discusses related work. Section III states the problem and summarizes the challenges and main results. Section IV presents a motivating example and provides conditions of identifiability. Sections V and VI present in detail the framework and mechanisms in the cases of trees and general topologies, respectively. Section VII concludes the paper.

II. RELATED WORK

Network Tomography. The term network tomography typically refers to a family of problems that aim at inferring internal network characteristics from measurements at the edge of the network. Internal characteristics of interest may include link-level parameters (such as loss and delay metrics) or the network topology. Another type of tomography problems aims at inferring path-level traffic intensity (e.g. traffic matrices) from link level measurements [10]. Our paper focuses on inferring the loss rates of internal links using active end-to-end measurements and assuming that the topology is known. Therefore it is related to the literature on loss tomography, part of which is discussed below.

Caceres et al. considered a single multicast tree with a known topology and inferred the link loss rates from the receivers' observations [2]. In particular, they developed a low-complexity algorithm to compute the maximum-likelihood estimator, by taking into account the dependencies introduced by the tree hierarchy to factorize the likelihood function and eventually compute the MLE in a recursive way. Throughout this paper, we refer to the MLE estimator for a multicast tree, developed in [2], as MINC, and we build on it. Bu et al. used multiple multicast trees to cover a general topology and proposed an EM algorithm for link loss rate estimation [3]. Follow-up approaches have been developed for unicast probes [5], [6], joint inference of topology and loss rates [4], adaptive tomography and delay inference [11]. The above list of references is not comprehensive. Good surveys of network tomography can be found in [1], [12].

Active vs. Passive Tomography. Tomography can be based either on active (generating probe traffic) or on passive (monitoring traffic flows and sampling existing traffic) measurements. Passive approaches have been most commonly used for estimating path level information, in particular, origin-destination traffic matrices, from data collected at various nodes of the network [10]. This approach and problem statement are well-suited for the needs of a network provider. For the problem of inferring link loss rates, active probes are typically used and information about individual packets received or lost is analyzed at the edge of the network. This approach is better suited for end-users that do not have access to the network. However, there are also papers that study link loss inference by using existing traffic flows to sample the state of the network [13], [14]. Once measurements have been collected following either of the two methods, statistical inference techniques are applied to determine network characteristics that are not directly observed.

The passive approach has the advantage that it does not impose additional burden on the network and that it measures the actual loss experienced by real traffic. However, it must also ensure that the characteristics of the traffic (e.g., TCP) does not bias the sample. In the active approach, one has more control over designing the probes, which can thus be optimized for efficient estimation. The downside is that we inject measurement traffic that may increase the load of the network, may be treated differently than regular traffic, or may even be dropped e.g., due to security concerns.

Network Coding and Inference. An extensive body of work on network coding [9], [15] has emerged after the seminal work of Ahlswede et al. [7] and Li et al. [8]. The main idea in network coding is that, if we allow intermediate nodes to not only forward but also combine packets, we can realize significant benefits in terms of throughput, delay, and robustness of distributed algorithms. Within this large body of work, closer to ours are a few papers that leverage the headers of network coded packets for passive inference of properties of a network. In [16], Ho et al. showed how information contained in network codes can be used for passive inference of possible locations of link failures or losses. In [17], Sharma et al. considered random intra-session network coding and showed that nodes can passively infer their upstream network topology, based on the headers of the received coded packets they observe (which play essentially the role of probes). The main idea is that the transfer matrix (i.e., the linear transform from the sender to the receiver) is distinct for different networks, with high probability. All possible transfer matrices are enumerated, and matched to the observed input/output, and a large finite field is used to ensure that all topologies remain distinguishable. An extended version of this work to erroneous networks is provided by Yao et al. in [18], where different (ergodic or adversarial) failures lead to different transfer functions. The approach in [17], [18] has the advantage of keeping the measurement bandwidth low (not higher than the transmission of coefficients, which is anyway required for data transfer with network coding) and the disadvantage of high complexity. In [19], Jafarisiavoshani et al. considered peer-to-peer

systems and used subspace nesting structures to passively identify local bottlenecks. Similarly to these papers, we leverage network coding operations for inference; in contrast to these papers, which use the headers of network-coded packets for passive inference of topology, we use the contents of active probes for inference of link loss rates.

Our Work. We were the first to make the connection between tomography and network coding capabilities. In [20], we introduced the basic idea of leveraging network coding capabilities to improve network monitoring. In [21], we studied link loss estimation in tree topologies. In [22], we extended the approach to general graphs. In [23], we built on [2] and we provided the MLE's of link loss rates in multiple-source tree topologies with network coding. This paper integrates ideas from these preliminary conference papers into a common framework, and extends them by a more in-depth analysis of identifiability, routing, estimation and code design.

Our approach is active in that probes are sent/received from/to the edge of the network and observations at the receivers are used for statistical inference. Intermediate nodes forward packets using unicast, multicast and simple coding operations. However, the operations at the intermediate nodes need to be set-up once, fixed for all experiments and be known for inference. Therefore, our approach requires more support from the network than traditional tomography, for the benefit of more accurate/efficient estimation. Our methods may also be applicable to passive tomography, where instead of sending specialized probes, one can view the coding coefficients on a network coded packet as the “probe”, thus overloading them with both communication and tomographic goals, as it is the case in [17], [18]. In this paper, we focus exclusively on the tomographic goals by taking an active approach, *i.e.*, by sending, collecting and analyzing specialized probes for tomography.

III. PROBLEM STATEMENT

A. Model and Definitions

1) *Network and Monitoring Scheme:* We consider a network represented as a graph $G = (V, E)$, where V is the set of nodes and E is the set of edges corresponding to *logical links*¹. We use the notation $e = AB$ for the link e connecting vertex A to vertex B . We assume that G is a directed graph without self-loops, and there is a loss rate associated with every edge in G . In general, the loss rates in the two directions of an edge can be different, as it is the case on the Internet, *e.g.*, due to different congestion levels. The topology $G = (V, E)$ is assumed to be known.

We assume that packet loss on a link $e \in E$ is i.i.d Bernoulli with probability $0 \leq \bar{\alpha}_e < 1$, where $\bar{\alpha}_e = 1 - \alpha_e$, and α_e is the success probability of link e . Losses are assumed to be independent across links. Let $\alpha = (\alpha_e)_{e \in E}$ be the vector of the link success probabilities². In loss tomography, we are interested in estimating all or a subset of the parameters in α ³. We use more specific notations for the case of tree topologies, as we explain in Section V-B.1.

A set S of $|S| = M$ source nodes in the periphery of the network can inject probe packets, while a set R of $|R| = N$ receivers can collect such packets. Several problem variations in the choice of sources and receivers are possible, and we will discuss the following in this paper: (i) the set of sources and the set of receivers are given and fixed; (ii) a set of nodes that can act as either sources or receivers is given (and we can select among them); (ii) we are allowed to select any node to act as a source or a receiver. We assume that intermediate nodes are equipped with unicast, multicast and network coding capabilities. Probe packets are routed and coded inside the network following specific paths and according to specified coding operations. The routes selected and the operations the intermediate nodes perform are part of the design of the tomography scheme: they are chosen once at set-up time and are kept the same throughout all experiments; all the operations of intermediate nodes are known during estimation.

In general, a probe packet is a vector of M symbols, with each symbol being in a finite field F_q . This includes as special cases: scalar network coding (for $M = 1$), operations over binary vectors (for $q = 2$), and more generally, vector network coding (for $M > 1$)⁴. In one *experiment*, we send probes from all sources and we collect probes at the receivers: each source $S_i \in S$ injects one probe packet x_i in the network, and each receiver $R_j \in R$ receives one probe X_j . The observations at all receivers R is a vector $X_{(R)} = (X_1, X_2, \dots, X_N)$ in the space $\Omega = (F_q^M)^N$. For a given set of link success probabilities $\alpha = (\alpha_e)_{e \in E}$, the probability distribution of all observations $X_{(R)}$ will be denoted by P_α . The probability mass function for a single observation $x \in \Omega$ is $p(x; \alpha) = P_\alpha(X_{(R)} = x)$.

To estimate the success rates of links, we perform a sequence of n independent experiments. Let $n(x)$ denote the number of probes for which the observation $x \in \Omega$ is obtained, where $\sum_{x \in \Omega} n(x) = n$. The probability of n independent observations

¹A logical link results from combining several consecutive physical links into a single link. This results in a graph G where every intermediate vertex has degree at least three, and in-degree and out-degree at least one. This is a standard assumption in the tomography literature, which is imposed for identifiability purposes, as discussed after Definition 2.

²Note that the notation α refers to the vector of all success probabilities, and α_e refers to the success prob. of an individual edge e .

³This is clearly equivalent to inferring the loss probabilities $1 - (\alpha_e)$, as it is the case in many tomography papers.

⁴What is important is that a probe can take one of q^M possible values. We note, however, that there is an equivalence between operations with elements in a finite field and operations with vectors of appropriate length. *E.g.*, in [24], the multicast scenario was considered, and scalar network coding over a finite field of size 2^M was used equivalently to vector network coding over the space of binary vectors of length M . Thinking in terms of one of the aforementioned special cases is appropriate in special topologies, as we will see *e.g.* in tree and reverse tree topologies, where scalars and binary vectors are used, respectively.

x^1, \dots, x^n (each $x^t = (x_k^t)_{k \in R}$) is:

$$p(x^1, x^2, \dots, x^n; \alpha) = \prod_{t=1}^n p(x^t; \alpha) = \prod_{x \in \Omega} p(x; \alpha)^{n(x)} \quad (1)$$

It is convenient to work with the log-likelihood function, which calculates the logarithm of this probability:

$$\mathcal{L}(\alpha) = \log p(x^1, \dots, x^n; \alpha) = \sum_{x \in \Omega} n(x) \log p(x; \alpha) \quad (2)$$

The goal is to use the observations at the receivers, the knowledge of the network topology, and the knowledge of routing/coding scheme to estimate the success rates of internal links of interest. We may be interested in estimating the success rate on a subset of links, or on all links.

We make two assumptions, which are both realistic in practice and standard in the tomography literature:

- The probability of loss $\bar{\alpha}_i$ on a link i is not 1, *i.e.*, $\bar{\alpha}_i \in [0, 1)$. This ensures that the log-likelihood function is well-defined and differentiable.
- We perform sufficient measurements so that, each observation $x \in \Omega$ at the receivers occurs at least once, *i.e.*, $n(x) > 0$. This ensures that no term in the likelihood function becomes a constant (due to a zero exponent).

Definition 1: A monitoring scheme for a given graph G refers to a set of M source nodes, a set of N receivers, a set of paths that connect the sources to the receivers, the probe packets that sources send, and the operations intermediate nodes perform on these packets.

We use the notion of link identifiability as it was defined in [2] (Theorem 3, Condition (i)):

Definition 2: A link e is called *identifiable* under a given monitoring scheme iff: $\alpha_e, \alpha'_e \in (0, 1]^{|E|}$ and $P_\alpha = P_{\alpha'}$ implies $\alpha_e = \alpha'_e$.

To illustrate the concept, consider two consecutive links $e_1 = AB$ and $e_2 = BC$ in a row, where node B has degree 2, and is neither a source nor a receiver. These links are not identifiable, as Eq.(3) would only allow us to identify the value of the product $\alpha_{e_1} \alpha_{e_2}$ and thus would lead to an infinite number of solutions. This is because, it is not possible to distinguish whether a packet gets dropped on link e_1 or e_2 . Note, however, that the case of having two links in a row is ruled out by our assumption of working on a graph with logical links (all vertices in the graph have degree three or greater). Another case that e_1, e_2 are not identifiable, and which is possible to happen even on a graph with logical links, is when the two links are crossed by exactly the same set of paths.

Identifiability is not only a property of the network topology, but also depends on the monitoring scheme. One of the main goals of the monitoring scheme design is to maximize the number of identifiable links. However, our definition of identifiability does not depend on the estimator employed. Essentially, identifiability depends on the probability distribution P_α and on whether this uniquely determines α .

2) *Estimation:* The maximum likelihood estimator (MLE) $\check{\alpha}$ identifies the parameters $(\alpha_e)_{e \in E}$ that maximize the probability of the observations $\mathcal{L}(\alpha)$:

$$\check{\alpha} = \operatorname{argmax}_{\alpha \in (0, 1]^{|E|}} \mathcal{L}(\alpha) \quad (3)$$

Candidates for the MLE are solutions $\hat{\alpha}$ of the *likelihood equation*:

$$\frac{\partial \mathcal{L}}{\partial \alpha_e}(\alpha) = 0, \quad e \in E \quad (4)$$

We can compute the MLE for tree networks as we see in Section V-B. However, it becomes computationally hard for large networks; this creates the need for faster algorithms that provide good approximate performance in practice.

To measure the per link estimation accuracy we will use the mean-squared error (MSE): $\text{MSE} = E(|\alpha_e - \hat{\alpha}_e|^2)$. In order to measure the estimation performance on all links $e \in E$, we need a metric that summarizes all links. We use an entropy measure ENT that captures the residual uncertainty. Since we expect the scaled estimation errors to be asymptotically Gaussian (similar to the case in [2]), we define the quality of the estimation across all links as

$$\text{ENT} = \sum_{e \in E} \log (E[\hat{\alpha}_e - \alpha_e]^2), \quad (5)$$

which is a shifted version of the entropy of independent Gaussian random variables with the given variances [25]. If the entire error covariance matrix \mathcal{R} is available, then we can compute the metric as $\text{ENT} = \log \det \mathcal{R}$, which captures also the correlations among the errors on different links. The metric ENT defined above captures only the diagonal elements of \mathcal{R} , *i.e.*, the MSE for each link independently of the others.

In some cases, we will approximate the error covariance matrix \mathcal{R} using the Fisher information matrix \mathcal{I} . Under mild regularity conditions (see for example Chapter 7 in [26]), the scaled asymptotic covariance matrix of the optimal estimator is

lower-bounded by the Cramer-Rao bound \mathcal{I}^{-1} . The Fisher information matrix \mathcal{I} is a square matrix with element $\mathcal{I}_{p,q}$ defined as

$$\mathcal{I}_{p,q}(\alpha) = -E \left[\frac{\partial}{\partial \alpha_p} \log p(X_{(R)}; \alpha) \frac{\partial}{\partial \alpha_q} \log p(X_{(R)}; \alpha) \right] \quad (6)$$

where α_p, α_q are the success probabilities of two links. In particular, under the regularity conditions, the MLE is asymptotically efficient, *i.e.*, it asymptotically, in sample size achieves this lower bound.

B. Subproblems

Given a certain network topology, a monitoring scheme for loss tomography can be designed by solving the following subproblems in a sequential way.

- 1) **Identifiability:** For each link $e \in E$, derive conditions that the scheme should satisfy so that the edge is identifiable. Whether the goal is to maximize the number of identifiable edges, or measure the link success rate on a particular set of edges, the identifiability conditions will guide the routing and code design choices.
- 2) **Routing:** Select the sources and receivers of probe packets, the paths through which probes are routed and the nodes where they will be linearly combined.⁵ The design goals include minimizing the utilized bandwidth, and improving the estimation accuracy, while respecting the required identifiability conditions.
- 3) **Probe and Code Design:** Select the contents of the probes sent by the sources and the operations performed at intermediate nodes. The goal is to use the simplest operations and the smallest finite field, while ensuring that the identifiability conditions are met.
- 4) **Estimation Algorithm:** This is the algorithm that processes the collected probes at the receivers and estimates the link loss rates. The objective is low complexity with good estimation performance. There is clearly a tradeoff between the estimation error and the measurement bandwidth.

We note that these steps are *not* independent from each other. In fact, the design of routing, probe and code design needs to be done with identifiability and estimation in mind.

C. Main Results

In this paper, we propose a monitoring scheme for loss tomography in networks that have multicast and network coding capabilities. In Sections V and VI, we present our design for the cases of trees and *general* topologies, respectively. We evaluate all our schemes through extensive simulation results. Below we preview the main results, in each subproblem.

- 1) **Identifiability:** (1) We provide simple necessary and sufficient conditions for link identifiability. (2) We also prove a structural property, which we call *reversibility*: if a link is identifiable under a given monitoring scheme, it remains identifiable if we reverse the directionality of all paths and exchange the role of sources and receivers (which we call the *dual configuration*).
- 2) **Routing:** (1) For a given set of sources and receivers over an arbitrary topology, the problem of selecting a routing that meets the identifiability conditions while minimizing the employed bandwidth is NP-hard. We show that, when network coding is used, this problem can be solved in polynomial time. (2) Moreover, we demonstrate that the choice of sources and receivers affects the estimation accuracy. (3) Finally, we present heuristic orientation algorithms for general graphs, designed to achieve identifiability, small number of receivers and high estimation accuracy.
- 3) **Probe and Code Design:** (1) In trees, we show that binary vectors sent by the sources and deterministic code design with XOR operations at intermediate nodes are sufficient. (2) In general graphs, we need to use operations over higher finite fields. We provide bounds on the required alphabet size, we propose and evaluate deterministic code design.
- 4) **Loss Estimation:** (1) For the purpose of estimating link loss rates in a tree topology with network coding capabilities, we design a low-complexity method for computing the MLE. We build on the low-complexity ML estimator for a multicast tree, developed in MINC [2]. We derive the MLE function of its dual configuration, *i.e.*, a reverse multicast tree (with several sources and one receiver), which we call RMINC. RMINC has the same functional form as MINC, due to the reversibility property. We then use MINC and RMINC to simultaneously identify the loss rates of all links in the original tree. (2) For topologies other than trees, ML estimation is computationally intensive; we propose a number of heuristic algorithms, including belief propagation and subtree decomposition algorithms, and we evaluate their performance through simulation.

The use of network coding at intermediate nodes, in addition to unicast and multicast, has the potential to offer several benefits for loss tomography:

- increased number of identifiable links (*i.e.*, whose loss probability can be inferred from end-to-end measurements);

⁵Depending on the practical constraints, such flexibility may or may not be available. If one cannot choose the source/receiver nodes and/or routing, as it is the case in most of the tomography literature, then this step can be skipped. If one can choose some of these parameters, then this can lead to further optimization of identifiability and estimation accuracy.

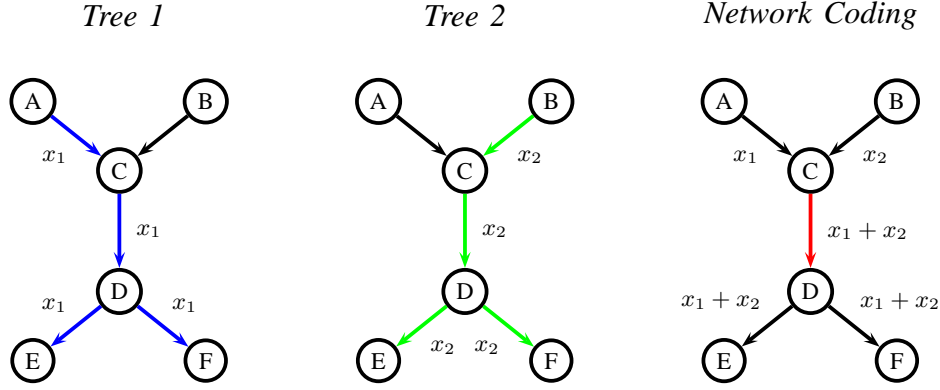


Fig. 1. Link loss monitoring for the basic 5-link topology. Nodes A and B are sources, E and F are receivers. Using multicast-based tomography, the topology can be covered using two multicast trees 1 and 2. Alternatively, the topology can be covered using coded packets, if node C can add (XOR) incoming packets.

- improved tradeoff between number of probes and estimation accuracy. (The intuition behind this benefit is two-fold: (a) when paths of probes overlap, the probes are coded together thus leading to exactly one probe crossing each link, and (b) each network coded probe brings more information, as it observes more than one path);
- the complexity of selecting probe paths for minimum cost monitoring of a general graph reduces from NP-hard to linear;
- the approach gracefully generalizes from trees to general topologies (*e.g.*, having the same identifiability conditions, using the same estimation algorithm, and avoiding the use of overlapping trees or paths) where its advantages are amplified.

IV. MOTIVATING EXAMPLE

In this section, (i) we present a motivating example to demonstrate the benefits of network coding in identifying the link loss rates; (ii) we derive the conditions of identifiability for a single link; and (iii) we discuss the identifiability of all links in the network.

Example 1: Consider the five-link topology depicted in Fig. 1. Nodes A and B send probes and nodes E and F receive them. Every link can drop a packet according to an i.i.d. Bernoulli distribution, with probability $\overline{\alpha}_e$, independently of other links. We are interested in estimating the success probabilities in all links, namely α_{AC} , α_{BC} , α_{CD} , α_{DE} , and α_{DF} .

The traditional multicast-based tomography approach would use two multicast trees rooted at nodes A and B and ending at E and F . This approach is depicted in Fig. 1-(a) and (b). At each experiment, source A sends packet x_1 and source B sends packet x_2 . The receivers E and F infer the link loss rates by keeping track of how many times they receive packets x_1 and x_2 . Note that, due to the overlap of the two trees, for each experiment, links CD , DE , and DF are used twice, leading to inefficient bandwidth usage. Moreover, from this set of experiments, we cannot calculate α_{CD} , and thus edge CD is not identifiable. Indeed, by observing the outcomes of experiments on each multicast tree, we cannot distinguish whether packet x_1 is dropped on edge AC or CD ; similarly, we cannot distinguish whether packet x_2 is dropped on edge BC or CD . (Note that if we restricted ourselves to unicast only, four unicast probes from A, B to E, F would be needed to cover all five links. Not only would the problems of identifiability and overlap of probe paths still be present but they would be further amplified.)

If network coding capabilities are available, they can help alleviate these problems. Assume that the intermediate node C can combine incoming packets before forwarding them to outgoing links. A sends to C a probe packet with payload that contains the binary string $x_1 = [1\ 0]$. Similarly, node B sends probe packet $x_2 = [0\ 1]$ to node C . If node C receives only x_1 or only x_2 , then it just forwards the received packet to node D ; if C receives both packets x_1 and x_2 , then it creates a new packet, with payload their linear combination $x_3 = [1\ 1]$, and forwards it to node D ; more generally, $x_3 = x_1 \oplus x_2$, where \oplus is the bit-wise XOR operation. Node D multicasts the incoming packet x_3 to both outgoing links DE and DF . The flow of packets in this experiment is shown in Fig. 1(c). In every experiment, probe packets (x_1, x_2) are sent from A, B and may or may not reach E, F , depending on the state of the links. Observe that with the network coding approach, link CD becomes identifiable. Moreover, we have avoided the overlap of probes on link CD during each experiment.

Table I lists the 10 possible observed outcomes, the state of links that leads to a particular outcome, the probability $p_i, i = 0, \dots, 9$ of observing this outcome, and the number of times $n_i, i = 0, \dots, 9$ we observe this outcome in a sequence of n independent experiments. The probability of observing an outcome, p_i , can be computed from the success probabilities $\alpha =$

#	Is link working (1) or not (0)?						Original (5-link) Tree actual probes received at:		Prob.	#times	Reduced Multicast Tree observations			Reduced Reverse Multicast Tree	
	AC	BC	CD	DE	DF		E	F	P_α		E	F	P_α^m	EF	P_α^r
1	Multiple possible events						-	-	p_0	n_0	0	0	p_0	[0, 0]	p_0
2	1	0	1	1	0		x_1	-	p_1	n_1	1	0	$p_1 + p_2 + p_3$	[1, 0]	$p_1 + p_4 + p_7$
3	0	1	1	1	0		x_2	-	p_2	n_2				[0, 1]	$p_2 + p_5 + p_8$
4	1	1	1	1	0		$x_1 \oplus x_2$	-	p_3	n_3				[1, 1]	$p_3 + p_6 + p_9$
5	1	0	1	0	1		-	x_1	p_4	n_4	0	1	$p_4 + p_5 + p_6$	[1, 0]	$p_1 + p_4 + p_7$
6	0	1	1	0	1		-	x_2	p_5	n_5				[0, 1]	$p_2 + p_5 + p_8$
7	1	1	1	0	1		-	$x_1 \oplus x_2$	p_6	n_6				[1, 1]	$p_3 + p_6 + p_9$
8	1	0	1	1	1		x_1	x_1	p_7	n_7	1	1	$p_7 + p_8 + p_9$	[1, 0]	$p_1 + p_4 + p_7$
9	0	1	1	1	1		x_2	x_2	p_8	n_8				[0, 1]	$p_2 + p_5 + p_8$
10	1	1	1	1	1		$x_1 \oplus x_2$	$x_1 \oplus x_2$	p_9	n_9				[1, 1]	$p_3 + p_6 + p_9$

TABLE I

THE 10 LEFTMOST COLUMNS OF THIS TABLE REFER TO THE 5-LINK TOPOLOGY SHOWN IN FIG.1(C). THEY SHOW THE POSSIBLE PAIRS OF PROBES COLLECTED (*i.e.*, THE OBSERVATIONS $x \in \Omega$) AT RECEIVERS E AND F , THEIR PROBABILITIES P_α , AND THE NUMBER OF TIMES n_i EACH OBSERVATION OCCURRED. THESE OBSERVATIONS DEPEND ON THE COMBINATION OF LOSS (0) AND SUCCESS (1) ON THE FIVE LINKS, WHICH HAPPEN W.P. α . THE REMAINING RIGHTMOST COLUMNS SHOW HOW THE SAME PROBES CAN BE INTERPRETED AS OBSERVATIONS AT THE RECEIVER(S) OF THE REDUCED TOPOLOGIES, NAMELY THE MULTICAST TREE AND THE REVERSE MULTICAST TREE, AS WE ARE GOING TO DESCRIBE IN SECTION V-B.3; AND THEIR CORRESPONDING PROBABILITIES.

$(\alpha_{AC}, \alpha_{BC}, \alpha_{CD}, \alpha_{DE}, \alpha_{DF})$ of the five links (AC, BC, CD, DE and DF respectively). *E.g.*, for outcomes 1-4:

$$\begin{aligned}
p_0 &= 1 - p_1 - \dots - p_9 = 1 - (1 - \bar{\alpha}_{AC}\bar{\alpha}_{BC})\alpha_{CD}(1 - \bar{\alpha}_{DE}\bar{\alpha}_{DF}) \\
p_1 &= \alpha_{AC}\bar{\alpha}_{BC}\alpha_{CD}\alpha_{DE}\bar{\alpha}_{DF} \\
p_2 &= \bar{\alpha}_{AC}\alpha_{BC}\alpha_{CD}\alpha_{DE}\bar{\alpha}_{DF} \\
p_3 &= \alpha_{AC}\alpha_{BC}\alpha_{CD}\alpha_{DE}\bar{\alpha}_{DF} \\
&\dots
\end{aligned} \tag{7}$$

and we can write similar expressions for the probabilities of the remaining observations. Thus, we can explicitly write down the probability distribution of the observations P_α .

In a sequence of $n = \sum_{i=0}^9 n_i$ independent experiments, the frequency of each event i is $\hat{p}_i \sim \frac{n_i}{n}$. After sending n independent probes, the log-likelihood function of the observations given the set of parameters (α_e) is: $\mathcal{L}(\alpha_{AC}, \alpha_{BC}, \alpha_{CD}, \alpha_{DE}, \alpha_{DF}) = \sum_{i=0}^9 n_i \cdot \log p_i(\alpha)$. The MLE would compute the α 's that maximize $\mathcal{L}(\alpha)$. \square

In general, we may be interested in estimating one of the α variables, some of them, or all five of them. In the next section, we discuss a single link, namely link CD . Note that the remaining four links can depict the equivalent paths connecting CD to the sources and receivers. In Section IV-B, we discuss the identifiability of all links.

A. Identifiability of One Link

Let us focus on a single link CD with success probability α_{CD} . Consider Fig. 2, which generalizes the motivating example of the previous section. Note that links other than CD can be viewed as summarizing paths: *e.g.*, AC could correspond to a path from A to C, possibly consisting of the concatenation of several links.

For a given a choice of sources and receivers and a coding scheme described in Section V-B.1, we want to translate the conditions for identifiability of link CD in Definition 2 to graph properties of the network. Our intuition is that a link CD is identifiable if C is a source, a coding point or a branching point, and D is a receiver, a coding point or a branching point. These are the structures depicted in Fig. 2, where we want to identify the link success rate associated with edge CD and interpret the remaining edges as corresponding to paths; *e.g.* AC could correspond to a path from A to C, possibly consisting of the concatenation of several links. The top two cases of Fig. 2 depict the simple cases where node C is a source, or node D is a receiver; the four bottom cases depict the case where C and D are coding or branching points.

To formalize this intuition, consider the following two conditions:

- **Condition 1:** At least one of the following holds:

- (a) $C \in S$.
- (b) There exist two edge-disjoint paths (X_1, C) and (X_2, C) that do not employ edge CD with distinct $X_1, X_2 \in S$.
- (c) There exist two paths (X_1, C) and (C, X_2) that do not employ CD with $X_1 \in S, X_2 \in R$.

- **Condition 2:** At least one of the following holds:

- (a) $D \in R$.
- (b) There exist two edge-disjoint paths (D, X_1) and (D, X_2) that do not employ edge CD with distinct $X_1, X_2 \in R$.
- (c) There exist two paths (X_1, D) and (D, X_2) that do not employ CD with $X_1 \in S, X_2 \in R$.

3-links, Multicast Tree 3-links, Reverse Multicast Tree

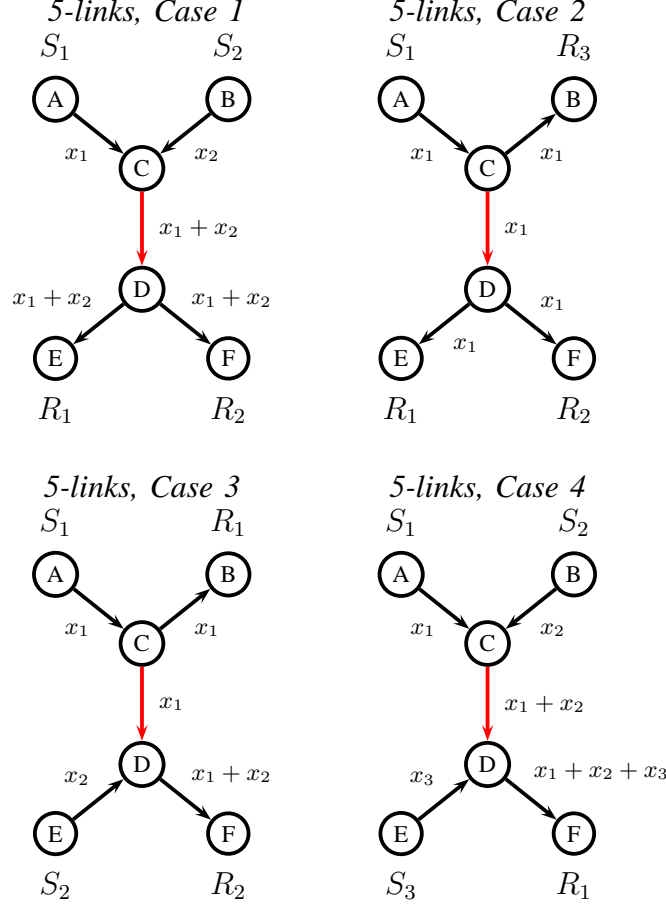
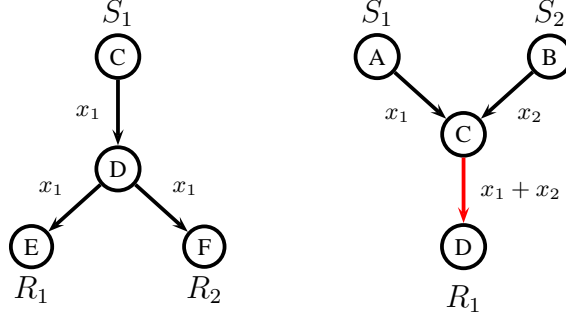


Fig. 2. **Configurations (i.e., combinations of Conditions 1 and 2) that allow us to identify the success rate of a single link (CD).** Recall that links other than CD, can correspond to paths with the same loss probability. The top of the figure shows a 3-link topology where C is a source (of a multicast tree) or D is a receiver (of a reverse multicast tree). The trivial case that C is a source and D is a receiver corresponds to a single link topology and is omitted here. The bottom of the figure shows a 5-link topology and four configurations (choices of sources and receivers) where neither C nor D are edge nodes and packets are sent and received at the edge nodes A, B, E and F. Case 1 is our familiar motivating example; Case 2 is similar to a single multicast tree rooted at A; Case 3 uses sources A and E and linear combinations whenever two flows meet; Case 4 does the same for sources A, B and E and is equivalent to an inverse multicast tree (with sink at F).

Theorem 4.1: For a given a choice of sources and receivers and a coding scheme described in Section V-B.1, link CD is identifiable if and only if both Conditions 1 and 2 hold.
The proof is provided in Appendix A.1.

B. Identifiability of All Links

In fact, we can identify *all* links at the same time. It is sufficient to ensure that each link is identifiable, according to the conditions of Theorem 4.1. This is true in all directed trees where each leaf node is either a source or a receiver, and each intermediate node satisfies the following mild conditions: (i) it has degree at least three (which is true in all logical links); (ii)

it has in-degree at least one (otherwise, the node should be a source); and (iii) it has out-degree at least one (otherwise the node should be a receiver).

Example 2: Table II lists which links are identifiable in the four bottom cases of Fig. 2, if we use our approach vs. if we use multicast tomography. All four configurations depict the same basic 5-link topology, but they differ in the choice of sources and receivers. Our approach is able to identify all links for any sets of sources and receivers. This is not always the case for the multicast tomography. \square

Case	Network Coding	Multicast Probes
1	all links	DE, DF
2	all links	all links
3	all links	AC, CB
4	all links	no links

TABLE II

IDENTIFIABLE LINKS IN THE FOUR CASES (DIFFERENT CHOICES OF SOURCES AND RECEIVERS, FOR THE SAME 5-LINK TOPOLOGY) DEPICTED AT THE BOTTOM OF FIG. 2.

V. TREE TOPOLOGIES

In this section, we consider tree topologies, and we describe our design choices in the four subproblems: we have already discussed identifiability in the previous section. Next, we describe routing in Section V-A, probe and code design in Section V-B.1 (operation of sources and intermediate nodes), and estimation algorithms in Sections V-B and V-C.

A. Routing, Selection of Sources and Receivers

Routing in trees is well defined: there exists a single path that connects a source to a receiver, through which probes flow. For a tree with L leaf nodes, some leaves act as sources S and the remaining leaves act as receivers $R = L \setminus S$. Intermediate nodes simply combine (XOR) the probes coming on all incoming links and forward (multicast) to all their outgoing links. This section looks at situations where we may have some freedom in the choice of the nodes that act as sources and receivers. If such flexibility is not available (as it is assumed in most tomography work), this step can be skipped. We study the effect of the selection of sources and receivers on estimation accuracy and we come up with some empirical guidelines for source selection, obtained through a number of examples and simulation scenarios.

Link loss tomography is essentially a parameter estimation problem, and different choices of sources and receivers lead to different estimators. That is, for a fixed number of probes, each topology leads to a different estimation accuracy; put differently, to achieve the same mean square error (MSE), we may need to use a different number of probes for each topology.

In Example 2, we saw that with network coding all links are identifiable, while if we use two multicast trees they are not. In Appendix B.2, we revisit the basic 5-link topology of Fig. 2 and we show that, even though with network coding links are identifiable for all four cases, the estimation accuracy differs depending on the number of sources and their relative position in the tree. This idea also applies to larger topologies. For example, in Appendix B.3, we consider a 9-link tree and we run simulations for different number and location of sources and we summarize the intuition obtained.

In general, the optimal selection of the number and location of sources depends on the network topology, the values of link-loss rates, and possibly the number of employed probes. This is currently an open problem.

B. Maximum Likelihood Estimation of Link Loss Rates

Ideally, it is desirable to use optimal (maximum likelihood) estimation to estimate the loss rates from the observations at the receivers. In the special case where the topology is a tree, and we can choose one source and several receivers so as to send probes across a *multicast tree*, an efficient ML estimator (MINC) has been designed in the pioneering paper [2]. We build on MINC, and we extend it to multiple-source trees with multicast and network coding capabilities [23]. We propose Alg. 1 in Section V-B.4, which provides an efficient way to compute the MLE for any link of interest, using the MLEs on a multicast and on a reverse multicast tree. This result can be used to infer the loss rates on all links at the same time.

Beyond the tree model we consider in Section V-B.1, there is no known computationally efficient algorithm to compute the MLE of all links at the same time. Therefore, we also propose three heuristic estimation algorithms and evaluate their performance through simulation. The first two (subtree decomposition and MINC-like heuristics, in Sections V-C.1 and V-C.2, respectively) are specific to trees, while the third one (belief propagation, in Section V-C.3) also applies to general graphs.

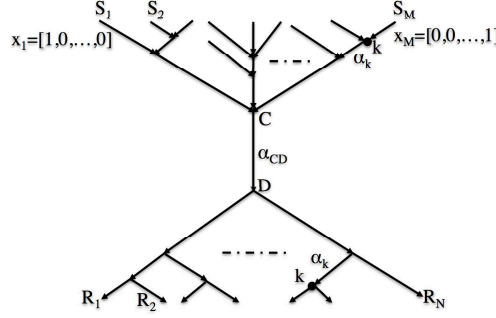


Fig. 3. A directed tree topology with multicast and network coding capabilities.

1) *Model and framework:* We first describe the model of tree networks for which we derive the MLE.

Logical Tree. We consider a tree topology, like the one depicted in Fig. 3, $G = (V, E)$ consisting of the set V of nodes and the set E of directed links. M leaf nodes, shown on top of the tree, act as sources of probe packets. The remaining N leaves, shown at the bottom of the tree, act as receivers. An intermediate node is either a coding point (with multiple incoming links and one outgoing link) or a branching point (with one incoming link and multiple outgoing links). For each node j , we denote the set of its parents (nodes with a link outgoing to j) by $f(j)$ and the set of its children (nodes with a link coming from j) by $d(j)$. The source nodes $S = \{S_1, \dots, S_M\}$ have no parent and the receiver nodes $R = \{R_1, \dots, R_N\}$ have no children. G, S, R are considered known and fixed throughout the measurements.

We note that the tree topology we consider has the property that all coding points are located above all branching points, which is a mild assumption⁶. It is also a logical tree (as described in Section III).

Operation of Sources. Each source S_i sends a probe packet x_i , which is a vector of length M in the following form:

$$x_i = \underbrace{[0, \dots, 0, 1, 0, \dots, 0]}_M, \quad i = 1, 2, \dots, M$$

Operation of Intermediate Nodes. Each coding point (bit-wise) XORs all packets it receives from its parents, and forwards the result to its child⁷. This very simple design effectively keeps the presence of each source orthogonal from every other source. This ensures versatility, in the sense that, no matter which probe packets get XOR-ed, they will not cancel each other out. For most practical purposes, this simple probe design is sufficient: a single IP packet can be up to 1500B (including the headers) and thus can accommodate roughly 12,000 probe sources (bits). In large networks, one can also spatially re-use probe packets by allocating the same probe packet to all sources whose packets do not meet. Finally, each branching point multicasts the packet it receives from its parent to all its children.

One can see that there will be a node after which $x_1 + x_2 + \dots + x_M$ flows through the network. We denote this node by C . Node C is the last coding point in the tree. Node C has P parents $f(C)_1, \dots, f(C)_P$, and only one child, which we denote by node D . Node D multicasts the packet it receives from node C to all its Q children $d(D)_1, \dots, d(D)_Q$.

We use the notation that $k < k', k, k' \in V$ when k is a descendant of k' , and that $k > k'$ when k is an ancestor of k' . Every node $k > C$ has multiple parents and only one child, while every node $k < D$ has one parent and multiple children. We are going to treat these two sets of nodes differently in the rest of Section V-B. We name any link of the tree that is above node C by its starting point, and we name any link that is below node D by its end point. In other words, link k denotes a link between nodes (k, j) if $k > C$ and $j > C$, while link k denotes a link between nodes (j, k) if $j < D$ and $k < D$.

Loss Model. As described in Section III, we model the loss rate of individual links by an i.i.d. Bernoulli process, independent across links. In particular, we use the following notation:

- A packet that traverses a link k above node C is lost with probability $\bar{\alpha}_k = 1 - \alpha_k$ and arrives at j with probability α_k .
- A packet that traverses a link k below node D is lost with probability $\bar{\alpha}_k = 1 - \alpha_k$ and arrives at k with probability α_k .
- Finally, we denote the loss rate of link CD by $\bar{\alpha}_{CD}$.

In general, we use the notation $\bar{\alpha} = 1 - \alpha$ for any quantity $0 < \alpha < 1$.

Let X_k denote the packet observed at node k , and let $X = (X_k)$, $k \in V$ denote the set of all X_k 's. X_k is a binary vector of length M . Its i^{th} element, $(X_k)_i$, represents the probe packet of source i : $(X_k)_i = 1$ indicates that the probe packet of source i reaches node k , and 0 that it does not. For the sources, $X_{S_i} = x_i$, thus $(X_{S_i})_i = 1$ and $(X_{S_{i'}})_{i'} = 0, \forall i' \neq i$. For any node

⁶Indeed: starting from an undirected tree, one can choose the sources so as to lead to a directed tree with this property. Note, however, that the tree model we consider for deriving the MLE contains all cases at the bottom of Fig. 2 except Case 3.

⁷We assume that the network is delay-free and all packet arrivals at a coding point are synchronized. Link delays only affect where the probe packets would meet.

$k \geq C$, if $(X_j)_i = 1$ for j a parent of k , $(X_k)_i = 1$ with probability α_j , and $(X_k)_i = 0$ with probability $\bar{\alpha}_j$, independently for all the parents of k . For any node $k \leq D$, if $X_k = [0, 0, \dots, 0]$ (the all-zero vector), then $X_j = [0, 0, \dots, 0]$, for the children j of k (and hence for all descendants of k). If $X_k \neq [0, 0, \dots, 0]$, then for j a child of k , $X_j = X_k$ with probability α_j , and $X_j = [0, 0, \dots, 0]$ with probability $\bar{\alpha}_j$, independently for all the children of k .

Data, Likelihood, and Inference. As described in Section III-A, in each experiment, one probe is dispatched from each source. The outcome of a single experiment is a record of whether or not each source probe was received at each receiver, which is the set of vectors X_k observed at receiver $k \in R$. It is denoted by $X_{(R)} = (X_k)_{k \in R}$ and is an element of the space $\Omega = \{[\dots, 0, 1, \dots]\}^N$ of all such outcomes. For a given set of link probabilities $\alpha = (\alpha_k)_{k \in V \setminus \{C, D\}} \cup \alpha_{CD}$, the distribution of the outcomes $X_{(R)}$ on Ω will be denoted by P_α . The probability mass function for a single outcome $x \in \Omega$ is $p(x; \alpha) = P_\alpha(X_{(R)} = x)$.

We perform n experiments. Let $n(x)$ denote the number of probes for which the outcome x is obtained. The probability of n independent observations x^1, \dots, x^n (each $x^t = (x_k^t)_{k \in R}$) is:

$$p(x^1, \dots, x^n; \alpha) = \prod_{t=1}^n p(x^t; \alpha) = \prod_{x \in \Omega} p(x; \alpha)^{n(x)} \quad (8)$$

Our task is to estimate α using maximum likelihood, from the data $(n(x))_{x \in \Omega}$. We work with the log-likelihood function:

$$\mathcal{L}(\alpha) = \log p(x^1, \dots, x^n; \alpha) = \sum_{x \in \Omega} n(x) \log p(x; \alpha) \quad (9)$$

The MLE of the loss rates $\check{\alpha}$ is the α that maximizes $\mathcal{L}(\alpha)$:

$$\check{\alpha} = \operatorname{argmax}_{\alpha \in (0,1]^{|E|}} \mathcal{L}(\alpha) \quad (10)$$

2) *The Likelihood Equation and its Solution:* Candidates for the MLE are solutions $\hat{\alpha}$ of the *likelihood equation*:

$$\frac{\partial \mathcal{L}}{\partial \alpha_k}(\alpha) = 0, \quad k \in V \quad (11)$$

We need to define some additional variables to compute the MLEs. For each node $k \geq D$, let $\Omega^r(k)$ be the set of outcomes $x \in \Omega$ such that $(x_a)_j \neq 0$ for at least one source $j \in S$ that is an ancestor of k and for any arbitrary set of receivers $\{a\} \subset R$. Let $\gamma_k^r = \Gamma_k^r(\alpha) = P_\alpha[\Omega^r(k)]$; an estimate of γ_k^r can be computed from:

$$\hat{\gamma}_k^r = \sum_{x \in \Omega^r(k)} \hat{p}(x), \quad \text{where} \quad \hat{p}(x) = \frac{n(x)}{n} \quad (12)$$

is the observed proportion of experiments with outcome x . γ_k^r shows the probability of the set of outcomes $\Omega^r(k)$ in which link k has definitely worked. Note that link k may have worked for some other outcomes as well, but they are not included in $\Omega^r(k)$. Also note that γ_k^r can be directly estimated from the observations at the receivers.

For each node $k \leq C$, we define $\Omega^m(k)$ to be the set of outcomes $x \in \Omega$ such that $x_j \neq [0, 0, \dots, 0]$ for at least one receiver $j \in R$ which is a descendant of k . Let $\gamma_k^m = \Gamma_k^m(\alpha) = P_\alpha[\Omega^m(k)]$; an estimate of γ_k^m is:

$$\hat{\gamma}_k^m = \sum_{x \in \Omega^m(k)} \hat{p}(x) \quad (13)$$

γ_k^m is the probability of the outcomes $\Omega^m(k)$ in which link k has definitely worked; it can be directly estimated from the observations at the receivers. Our goal is to compute $\hat{\alpha}$ from $\hat{\gamma} = (\hat{\gamma}_k^r \cup \hat{\gamma}_k^m)_{k \in V}$.

Special Case (i): Multicast Tree (MINC). If $M = 1$, the general model turns into a multicast tree with a single source, which is the case considered in [2]. We represent the source node by $0 \in V$. Each node j other than the source node, has one parent $f(j)$, and a set $d(j)$ of children. We denote the link loss rates by $\bar{\alpha}_k$, where k is the end point. We simply assume that $\alpha_0 = 1$.

The outcome of each experiment is $X_{(R)} = (X_k)_{k \in R}$, where each X_k is a single binary value (instead of a binary vector of length M in the general case), corresponding to whether the source probe is observed at each receiver $k \in R$ or not. The state space of the observations $X_{(R)}$ is $\Omega = \{0, 1\}^N$. We say that a link k is at level $l^m(k)$ if there is a chain of $l^m(k)$ ancestors $k < f(k) < f^2(k) < \dots < f^{l^m(k)}(k) = 0$ leading back to the source.

Only $\Omega^m(k)$ is used for each node k in the multicast tree; it is the set of outcomes $x \in \Omega$ where $x_j = 1$ for at least one receiver $j \in R$ that is a descendant of k . The definition of γ_k^m is like before.

The MLE for the multicast tree has been computed in [2]. Let $A_k^m = \prod_{i=0}^{l^m(k)} \alpha_{f^i(k)}$ show the probability that the path from the source to node k works, which we denote by $P(Y_{0 \rightarrow k} = 1)$. Its estimate \hat{A}_k^m can be computed as follows. For the source

node, $\hat{A}_0^m = 1$, for the leaf nodes $k \in R$, $\hat{A}_k^m = \hat{\gamma}_k^m$, and for all other nodes $k \in V \setminus \{0, R\}$, \hat{A}_k^m is the unique solution in $(0, 1]$ of:

$$1 - \frac{\hat{\gamma}_k^m}{\hat{A}_k^m} = \prod_{j \in d(k)} \left(1 - \frac{\hat{\gamma}_j^m}{\hat{A}_j^m}\right) \quad (14)$$

$\hat{\alpha}_k$ can then be computed from $\hat{\gamma}_k^m$, i.e., $\hat{\alpha} = \Gamma^{m-1}(\hat{\gamma}^m)$, as follows:

$$\hat{\alpha}_k = \frac{\hat{A}_k^m}{\hat{A}_{f(k)}^m}, \quad k \in V \setminus \{0\} \quad (\hat{\alpha}_0 = 1) \quad (15)$$

We refer to Eq.(15) as MINC in the rest of the paper.

Note. Eq.(14) is obtained from the following relations, after some computations in [2], which we repeat here for completeness. Let $\beta_k^m = P[\Omega^m(k) | X_{f(k)} = 1]$ denote the conditional probability of $\Omega^m(k)$ given that $f(k)$ has observed something. Failure can be due to either $\bar{\alpha}_k$ (failure of link k), or all paths towards the destinations failing. The β_k^m obey the following recursion:

$$\bar{\beta}_k^m = \bar{\alpha}_k + \alpha_k \prod_{j \in d(k)} \bar{\beta}_j^m, \quad k \in V \setminus R \quad (16)$$

$$\beta_k^m = \alpha_k, \quad k \in R \quad (17)$$

Eq.(14) then follows from the following relation between α and γ^m :

$$\gamma_k^m = \beta_k^m \prod_{i=1}^{l^m(k)} \alpha_{f^i(k)} \quad (18)$$

Special Case (ii): Reverse Multicast Tree (RMINC). If $N = 1$, the general model turns into a reverse multicast tree with a single receiver, which we denote by $0 \in V$. Each node j other than 0 has one child $d(j)$, and a set $f(j)$ of parents. We denote link loss rates by $\bar{\alpha}_k$, where k is the starting point. We assume that $\alpha_0 = 1$.

The outcome of each experiment, X_R , is a binary vector of length M . Each of its elements, $(X_R)_i$, represents whether the probe packet of source i is observed at the receiver or not. The state space of the observations X_R is $\Omega = \{0, 1\}^M$. We say that a link k is at level $l^r(k)$ if there is a chain of $l^r(k)$ descendants $k > d(k) > d^2(k) \cdots > d^{l^r(k)}(k) = 0$ leading down to the receiver.

Only $\Omega^r(k)$ is used for each node k in the reverse multicast tree; it is the set of outcomes $x \in \Omega$ where $x_j = 1$ for at least one source $j \in S$ that is an ancestor of k . The definition of γ_k^r is like before.

The MLE for the reverse multicast tree is similar to the multicast tree. Let $A_k^r = \prod_{i=0}^{l^r(k)} \alpha_{d^i(k)}$ show the probability that the path from node k to the receiver node works, which we denote by $P(Y_{k \rightarrow 0} = 1)$. Its estimate \hat{A}_k^r can be computed as follows. For the receiver node, $\hat{A}_0^r = 1$, for the source nodes $k \in S$, $\hat{A}_k^r = \hat{\gamma}_k^r$, and for all other nodes $k \in V \setminus \{S, 0\}$, \hat{A}_k^r is the unique solution in $(0, 1]$ of:

$$1 - \frac{\hat{\gamma}_k^r}{\hat{A}_k^r} = \prod_{j \in f(k)} \left(1 - \frac{\hat{\gamma}_j^r}{\hat{A}_j^r}\right) \quad (19)$$

We can then compute $\hat{\alpha}_k$ from $\hat{\gamma}_k^r$, i.e., $\hat{\alpha} = \Gamma^{r-1}(\hat{\gamma}^r)$, as follows:

$$\hat{\alpha}_k = \frac{\hat{A}_k^r}{\hat{A}_{d(k)}^r}, \quad k \in V \setminus \{0\} \quad (\hat{\alpha}_0 = 1) \quad (20)$$

We refer to Eq.(20) as RMINC in the rest of the paper.

Note. Eq.(19) results from the following relations. Let $\beta_k^r = P[\Omega^r(k) | Y_{d(k) \rightarrow 0} = 1]$ denote the conditional probability of $\Omega^r(k)$ given that the path from $d(k)$ to the receiver works. We have that:

$$\bar{\beta}_k^r = \bar{\alpha}_k + \alpha_k \prod_{j \in f(k)} \bar{\beta}_j^r, \quad k \in V \setminus S \quad (21)$$

$$\beta_k^r = \alpha_k, \quad k \in S \quad (22)$$

$$\gamma_k^r = \beta_k^r \prod_{i=1}^{l^r(k)} \alpha_{d^i(k)} \quad (23)$$

Comparison of MINC and RMINC. The reader will notice that the MLE for the multicast tree and the reverse multicast tree have the same functional form. This is a special case of the more general “reversibility” property, first observed in [22]. Indeed, there is a 1-1 correspondence between the observable outcomes in the two cases; furthermore, the corresponding

outcomes have the same probability, as a function of α_k 's, thus leading to the same MLE. In the following, we describe the reversibility property in more detail.

Reversibility – A Structural Property. Consider a tree topology $G = (V, E)$ with L leaf nodes, some of which act as sources S and the remaining ones, $R = L \setminus S$, act as receivers of probes. Routing from S to R is given (e.g., determined in the routing subproblem) and defines a direction on every link $e \in E$, along which probes flow.

Definition 3: We call the triplet (G, S, R) a *configuration*.

We define as dual the configuration that results from reversing the orientation of all links in the network, and from having the sources S become receivers, while the receivers R act as sources. More formally:

Definition 4: Consider the original configuration (G, S, R) . Consider the graph $G^d = (V, E^d)$ that has the same nodes but reversed edges, i.e., $e = (i, j) \in E$ iff $e^d = (j, i) \in E^d$ and success rate $\alpha_e^d = \alpha_e$, associated with every edge $e^d \in E^d$. Select sources $S^d = R$ and receivers $R^d = S$. We call the (G^d, S^d, R^d) the *dual configuration* of (G, S, R) .

For example, a multicast tree is the dual configuration of a reverse multicast tree (Cases 2 and 4 in Fig. 2). In Appendix B, we show that the dual configurations of Fig. 24(a) and Fig. 24(b) result in the same mean square error bound. In fact, a closer look reveals that not only the values but also the functional forms of these two ML estimators coincide. The following theorem generalizes this notion to general trees.

Theorem 5.1: Consider a configuration (G, S, R) with observations at the receivers Ω , and probability distribution $P_\alpha = \{p(x; \alpha), x \in \Omega\}$. Consider its dual configuration (G^d, R, S) , with observations Ω^d and probability distribution P_α^d . Then, there is a bijection between outcomes and their probabilities in the original ($x \in \Omega, p(x; \alpha)$) and in the dual configuration ($x^d \in \Omega^d, p(x^d; \alpha)$).

Proof: Let $G = (V, E)$ be the original tree graph, and G^d its dual. In every experiment, there exist $2^{|E|}$ possible error events, depending on which subset of the links fail. Observing the outcomes at the receivers corresponds to observing unions of events, that occur with the corresponding probability (e.g., as in the example of Table I). We show that for every observable outcome, that occurs with probability p in G , there exists exactly one observable outcome that occurs with the same probability in G^d and vice-versa. This establishes a bijection.

With every edge e of G , we can associate a set of sources $S(e) \subset V$ that flow through this edge, and a set of receivers $R(e) \subset V$ that observe the flow through e . Our main observation is that the pair $\{S(e), R(e)\}$ uniquely identifies e , i.e., no other edge has the same pair. In the dual configuration G^d , edge e is uniquely identified by the pair $\{R(e), S(e)\}$. If in G edge e fails while all other edges do not, the receivers $R(e)$ will not receive the contribution in the probe packets of the sources $S(e)$. If in G^d edge e fails while all other edges do not, the receivers $S(e)$ will not receive the contribution in the probe packets of the sources $R(e)$. Thus, there is a one-to-one mapping between these events. Using this equivalence, an observable outcome consisting of a union of events can be mapped to an observable outcome in the reverse tree. ■

Corollary 5.2: The maximum likelihood estimators for a configuration and its dual have the same functional form.

Proof: The bijection established above implies that a configuration and its dual have the same set of observable outcomes, with the same probabilities. Therefore, they have the same likelihood function and thus the same maximum likelihood estimator. ■

We note that this corollary establishes reversibility only for the maximum likelihood estimation. The performance of suboptimal algorithms may differ when applied to a configuration and its dual.

Application to Measuring Directional Networks. It is also important to note that the notion of dual configurations does *not* assume that the loss rates in both directions of a link are the same. Reversibility means that the two ML estimators for a configuration and its dual are described by the same function; however, the loss parameters we try to estimate (using the same estimator function) in the two directions may have different values. In fact, consider a tree with links that have different loss rates in the two directions. In this case, the reversibility property can be exploited to efficiently monitor all links and directions. Indeed, it is sufficient to send probes over only two configurations: the original and its dual.

Corollary 5.3: Consider a tree G with L leaves, where each leaf is either a source or a receiver. We are interested in measuring the loss rates in both directions for all links of the tree. Using network coding saves a factor of L in bandwidth used by probes, compared to the multicast tree approach.

Proof: Consider a tree configuration with L leaves. To measure the link loss rates in both directions for all edges of the tree, using the multicast approach, we need to use L multicast trees. Indeed, let $e = AC$ be the link adjacent to leaf $A \in L$, we can measure α_{AC} only if A is the root of the multicast tree. Using the network coding approach, for any choice of sources and receivers, we only need to perform two rounds of measurements: one on the network G and one on its dual G^d . ■

This corollary can also be interpreted as a tradeoff in directional measurement. We can either L -fold increase the measurement bandwidth (using multicast probes), or allow intermediate nodes to do linear combinations (network coding). The former option keeps intermediate nodes simple at the expense of using extra bandwidth. The latter option sends exactly one probe per link per experiment, but requires some operations from intermediate nodes.

3) *Maximum Likelihood Estimation of Loss Rates:* We now present how to “reduce” the original tree to a multicast and to a reverse multicast tree, and how to estimate α_{CD} . These intermediate results are then used in the MLE algorithm in Section V-B.4.

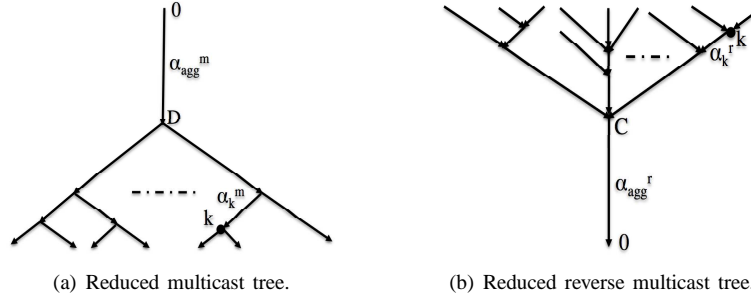


Fig. 4. Reducing the tree topology in Fig. 3 to a multicast tree and to a reverse multicast tree.

Reduction to a Multicast Tree (m). If we take the upper part of the original tree in Fig. 3 and consider it as an aggregate link, we obtain the reduced multicast tree in Fig. 4(a). The aggregate link agg^m summarizes the operation of all links above C and link CD . Node D receives a packet if at least one path from the sources to C works and link CD works. In other words, the success probability of the aggregate link, α_{agg}^m , depends on the paths from the sources to node C , and also link CD .

More formally, we map the outcomes $x \in \Omega$ of the original tree to the outcomes x^m of the multicast tree, as follows. Each x is a set of N binary vectors, each of length M , while each x^m is a single binary vector of length N . Any outcome x^m is obtained by taking a set of outcomes $\{x\}$, in all of which the same receivers have observed all-zero vectors and the same receivers have observed non-zero vectors, and by replacing each non-zero vector (that may contain any of the source probes x_1, x_2, \dots, x_M) by value 1, and by replacing each all-zero vector by value 0. *I.e.:*

$$\sum_{x_{R_t} \neq [0,0,\dots,0], x_{R_{t'}} = [0,0,\dots,0]} n(x) = n^m(x^m), x_{R_t}^m = 1, x_{R_{t'}}^m = 0, t, t' \in \{1, \dots, N\}, t \neq t' \quad (24)$$

If the original tree has link success rates α and an associated probability distribution of outcomes P_α , then the multicast tree is defined with parameters α^m and associated probability distribution P_α^m , such that:

$$\alpha_k^m = \alpha_k, k < D, \quad \alpha_{agg}^m = \alpha_{CD} \left(1 - \prod_{i=1}^P \bar{\beta}_{f(C)_i}^r\right) \quad (25)$$

P_α^m can be directly calculated from P_α , since each event in P_α^m is the union of a disjoint subset of events in P_α and has probability equal to the sum of probabilities of those events in P_α (such as the 5-link example in Table I).

Reduction to a Reverse Multicast Tree (r). Similarly, if we consider the lower part of the original tree in Fig. 3 as an aggregate link, we obtain the reduced reverse multicast tree in Fig. 4(b), with parameters α^r and associated probability distribution P_α^r , such that:

$$\alpha_k^r = \alpha_k, k > C, \quad \alpha_{agg}^r = \alpha_{CD} \left(1 - \prod_{j=1}^Q \bar{\beta}_{d(D)_j}^m\right) \quad (26)$$

The Relation Between the Two Reduced Trees.

Lemma 5.4: We have that: $\hat{\gamma}_C^r = \hat{\gamma}_D^m = 1 - \hat{p}([0,0,\dots,0])$.

The proof directly results from the definition of γ_D^m in the reduced multicast tree and γ_C^r in the reduced reverse multicast tree.

Estimating α_{CD} . The MLE of α_{CD} can be obtained from:

$$\hat{\alpha}_{CD} = \frac{\hat{A}_C^r \cdot \hat{A}_D^m}{\hat{\gamma}_C^r} = \frac{\hat{A}_C^r \cdot \hat{A}_D^m}{\hat{\gamma}_D^m} \quad (27)$$

The proof can be found in Appendix A.2.

4) *The Analysis of the MLE:* In this Section, we propose the MLE algorithm, we discuss its complexity, and we illustrate our results through the example tree topology in Fig. 1(c).

MLE Algorithm. Algorithm 1 computes the MLE of all link loss rates in the tree topology of Fig. 3; it proceeds in the following steps: (i) it computes $\hat{\alpha}_k$ for any link k below node D from the reduced multicast tree using Eq.(15); (ii) it computes $\hat{\alpha}_k$ for any link k above node C from the reduced reverse multicast tree using Eq.(20); and (iii) it computes $\hat{\alpha}_{CD}$ from Eq.(27). These are indeed the MLEs of the link loss rates, $\hat{\alpha}$, for the tree of Fig. 3.

Theorem 5.5: The estimates computed by Algorithm 1 are the MLEs of the link loss rates in the original tree topology in Fig. 3.

The proof of Theorem 5.5 relies on the following lemmas, whose proofs are provided in Appendix A.3. (Theorem 5.5 is then proved in Appendix A.4.)

Algorithm 1 Computing the MLE of all link loss rates in the original tree topology of Fig. 3.

```

1: for all links  $k$ , where  $k < D$  do
2:   Reduce the original tree to a multicast tree. Use MINC [2] (Eq.(15)) to compute the MLEs  $\hat{\alpha}_k^m$  and  $\hat{\alpha}_{agg}^m$ .
3:   Let  $\hat{\alpha}_k = \hat{\alpha}_k^m$ .
4: end for
5: for all links  $k$ , where  $k > C$  do
6:   Reduce the original tree to a reverse multicast tree. Use RMINC (Eq.(20)) to compute the MLEs  $\hat{\alpha}_k^r$  and  $\hat{\alpha}_{agg}^r$ .
7:   Let  $\hat{\alpha}_k = \hat{\alpha}_k^r$ .
8: end for
9: Use Eq.(27) to compute the MLE  $\hat{\alpha}_{CD}$ .

```

Lemma 5.6: The solutions of the likelihood equations of the original tree and the reduced multicast tree are related via: (i) $\hat{\alpha}_k = \hat{\alpha}_k^m$, $k < D$; and (ii) $\hat{\alpha}_{CD} = \hat{\alpha}_{agg}^m / (1 - \prod_{i=1}^P \bar{\beta}_{f(C)_i}^r)$.

Lemma 5.7: The solutions of the likelihood equations of the original tree and the reduced reverse multicast tree are related via: (i) $\hat{\alpha}_k = \hat{\alpha}_k^r$, $k > C$; and (ii) $\hat{\alpha}_{CD} = \hat{\alpha}_{agg}^r / (1 - \prod_{j=1}^Q \bar{\beta}_{d(D)_j}^m)$.

We note that the likelihood functions of the original tree and the reduced multicast (or reverse multicast) tree are different. What the aforementioned lemmas establish is that these likelihood functions are maximized for the same values of their common variables.

Complexity. Algorithm 1 is very efficient. In the first two steps, it calls MINC and RMINC. MINC (and thus RMINC) is known to be efficient by exploiting the hierarchy of the tree topology to factorize the probability distribution and recursively compute the estimates. The computation at each node is at worst proportional to the depth of the tree [2]. The last step, $\hat{\alpha}_{CD}$, uses the estimates $\hat{A}_k, \hat{\gamma}_k$ already computed in the first two steps.

Rate of Convergence of the MLE. We can provide the rate of convergence of $\hat{\alpha}$ to the true value α . The Fisher information matrix at α based on $X_{(R)}$ is obtained from $\mathcal{I}_{jk}(\alpha) = -E \frac{\partial^2 \mathcal{L}}{\partial \alpha_j \partial \alpha_k}(\alpha)$ [2]. We have that:

Theorem 5.8: $\mathcal{I}(\alpha)$ is non-singular, and as $n \rightarrow \infty$, $\sqrt{n}(\hat{\alpha} - \alpha)$ converges in distribution to $\mathcal{N}(0, \mathcal{I}^{-1}(\alpha))$.

The proof follows from the asymptotic properties of the MLEs [2], [27]. Therefore, asymptotically for large n , with probability $1 - \delta$ (for $1 - \delta$ confidence interval), $\hat{\alpha}_k$ lies between the points:⁸

$$\alpha_k \pm z_{\delta/2} \sqrt{\frac{\mathcal{I}_{kk}^{-1}(\alpha)}{n}} \quad (28)$$

Example 3: We now illustrate our results by revisiting the example 5-link tree topology in Fig. 1(c). Note that here, following the notation described in Section V-B.1, we use the notation $\alpha_A, \alpha_B, \alpha_E$, and α_F , for the four edge links in Fig. 1(c), instead of $\alpha_{AC}, \alpha_{BC}, \alpha_{DE}$, and α_{DF} respectively, which were used in Example 1.

Maximum Likelihood Estimator. The two source nodes A and B send probe packets $x_1 = [1, 0]$ and $x_2 = [0, 1]$ respectively. Ω consists of ten possible outcomes shown in Table I. Table I also shows the corresponding outcomes of the reduced multicast and reverse multicast trees. From Eq.(12) and Eq.(13), we have that:

$$\begin{aligned}
\hat{\gamma}_A^r &= \hat{p}_1 + \hat{p}_3 + \hat{p}_4 + \hat{p}_6 + \hat{p}_7 + \hat{p}_9 \\
\hat{\gamma}_B^r &= \hat{p}_2 + \hat{p}_3 + \hat{p}_5 + \hat{p}_6 + \hat{p}_8 + \hat{p}_9 \\
\hat{\gamma}_C^r &= \hat{\gamma}_D^m = \hat{p}_1 + \hat{p}_2 + \hat{p}_3 + \hat{p}_4 + \hat{p}_5 + \hat{p}_6 + \hat{p}_7 + \hat{p}_8 + \hat{p}_9 = 1 - \hat{p}_0 \\
\hat{\gamma}_E^m &= \hat{p}_1 + \hat{p}_2 + \hat{p}_3 + \hat{p}_7 + \hat{p}_8 + \hat{p}_9 \\
\hat{\gamma}_F^m &= \hat{p}_4 + \hat{p}_5 + \hat{p}_6 + \hat{p}_7 + \hat{p}_8 + \hat{p}_9
\end{aligned}$$

We then solve Eq.(14) for \hat{A}_k^m and Eq.(19) for \hat{A}_k^r , and then we find $\hat{\alpha}_A$ and $\hat{\alpha}_B$ from Eq.(20), $\hat{\alpha}_E$ and $\hat{\alpha}_F$ from Eq.(15), and $\hat{\alpha}_{CD}$ from Eq.(27), as follows:

$$\hat{\alpha}_A = \frac{\hat{\gamma}_A^r + \hat{\gamma}_B^r - \hat{\gamma}_C^r}{\hat{\gamma}_B^r}, \quad \hat{\alpha}_B = \frac{\hat{\gamma}_A^r + \hat{\gamma}_B^r - \hat{\gamma}_C^r}{\hat{\gamma}_A^r} \quad (29)$$

$$\hat{\alpha}_E = \frac{\hat{\gamma}_E^m + \hat{\gamma}_F^m - \hat{\gamma}_D^m}{\hat{\gamma}_F^m}, \quad \hat{\alpha}_F = \frac{\hat{\gamma}_E^m + \hat{\gamma}_F^m - \hat{\gamma}_D^m}{\hat{\gamma}_E^m} \quad (30)$$

$$\hat{\alpha}_{CD} = \frac{\hat{\gamma}_A^r \hat{\gamma}_B^r \hat{\gamma}_E^m \hat{\gamma}_F^m}{\hat{\gamma}_D^m (\hat{\gamma}_A^r + \hat{\gamma}_B^r - \hat{\gamma}_C^r) (\hat{\gamma}_E^m + \hat{\gamma}_F^m - \hat{\gamma}_D^m)} \quad (31)$$

Confidence Intervals. Fig. 5 shows $\mathcal{I}^{-1}(\alpha)$ for the confidence intervals in Eq.(28). We note that the confidence intervals for parameters $\hat{\alpha}$ can be obtained by inserting Eq.(29), Eq.(30), and Eq.(31) into Fig. 5. \square

⁸ $z_{\delta/2}$ denotes the number that cuts off an area $\delta/2$ in the right tail of the standard normal distribution.

$$\mathcal{I}^{-1}(\alpha) = \begin{pmatrix} \frac{\alpha_A \bar{\alpha}_A}{\alpha_B \alpha_{CD} (\alpha_E + \alpha_F - \alpha_E \alpha_F)} & \frac{\bar{\alpha}_A \bar{\alpha}_B}{\alpha_{CD} (\alpha_E + \alpha_F - \alpha_E \alpha_F)} & \frac{-\bar{\alpha}_A \bar{\alpha}_B}{\alpha_B (\alpha_E + \alpha_F - \alpha_E \alpha_F)} & 0 & 0 \\ \frac{\alpha_A \bar{\alpha}_B}{\alpha_{CD} (\alpha_E + \alpha_F - \alpha_E \alpha_F)} & \frac{\alpha_A \alpha_{CD} (\alpha_E + \alpha_F - \alpha_E \alpha_F)}{-\bar{\alpha}_A \bar{\alpha}_B} & \frac{\alpha_A (\alpha_E + \alpha_F - \alpha_E \alpha_F)}{-\bar{\alpha}_A \bar{\alpha}_B} & 0 & 0 \\ \frac{\alpha_B (\alpha_E + \alpha_F - \alpha_E \alpha_F)}{-\bar{\alpha}_A \bar{\alpha}_B} & \frac{-\bar{\alpha}_A \bar{\alpha}_B}{\alpha_A (\alpha_E + \alpha_F - \alpha_E \alpha_F)} & \frac{T_{33}^{-1}(\alpha)}{-\bar{\alpha}_B \bar{\alpha}_F} & \frac{-\bar{\alpha}_F \bar{\alpha}_F}{\alpha_F (\alpha_A + \alpha_B - \alpha_A \alpha_B)} & \frac{-\bar{\alpha}_F \bar{\alpha}_F}{\alpha_E (\alpha_A + \alpha_B - \alpha_A \alpha_B)} \\ 0 & 0 & \frac{\alpha_F (\alpha_A + \alpha_B - \alpha_A \alpha_B)}{-\bar{\alpha}_B \bar{\alpha}_F} & \frac{\alpha_{CD} \alpha_F (\alpha_A + \alpha_B - \alpha_A \alpha_B)}{\bar{\alpha}_E \bar{\alpha}_F} & \frac{\alpha_{CD} (\alpha_A + \alpha_B - \alpha_A \alpha_B)}{\bar{\alpha}_E \bar{\alpha}_F} \\ 0 & 0 & \frac{\alpha_E (\alpha_A + \alpha_B - \alpha_A \alpha_B)}{-\bar{\alpha}_B \bar{\alpha}_F} & \frac{\alpha_{CD} (\alpha_A + \alpha_B - \alpha_A \alpha_B)}{\bar{\alpha}_E \bar{\alpha}_F} & \frac{\alpha_{CD} \alpha_E (\alpha_A + \alpha_B - \alpha_A \alpha_B)}{\bar{\alpha}_E \bar{\alpha}_F} \end{pmatrix}$$

$$\mathcal{I}_{33}^{-1}(\alpha) = \frac{1}{\alpha_A \alpha_B \alpha_E \alpha_F (-\alpha_A \bar{\alpha}_B - \alpha_B) (-\alpha_E \bar{\alpha}_F - \alpha_F)} (-\alpha_{CD} (-\alpha_B \bar{\alpha}_B \alpha_E \alpha_F - \alpha_A^2 \bar{\alpha}_B \alpha_E (-1 + \alpha_B (2 + \alpha_{CD} (-\alpha_E \bar{\alpha}_F - \alpha_F))) \alpha_F$$

$$+ \alpha_A (-\alpha_E \alpha_F + \alpha_B^2 \alpha_E \alpha_F (-3 + \alpha_{CD} (\alpha_E + \alpha_F - \alpha_E \alpha_F))) + \alpha_B (-\alpha_F \bar{\alpha}_F + \alpha_E (-1 + 7\alpha_F - 3\alpha_F^2) + \alpha_E^2 (1 - 3\alpha_F + 2\alpha_F^2)))$$

Fig. 5. The inverse of the Fisher information matrix governing the confidence intervals for models in Eq.(28). Here, the order of the coordinates is $\alpha_A, \alpha_B, \alpha_{CD}, \alpha_E, \alpha_F$.

C. Heuristic Approaches for Loss Estimation

1) **Subtree Decomposition:** This algorithm partitions the tree into multicast subtrees separated by coding points. Each coding point virtually acts as a receiver for incoming flows and as a source for outgoing flows. As a result, each subtree will either have a coding point as its source, or will have at least one coding point as a receiver. In each subtree, we can then use the MLE estimator (MINC) proposed in [2].

Note that we can only observe packets received at the edge of the network but not at the coding points. However, we can still infer that information from the observations at the receivers downstream from the coding point. The fact that we infer observations of the coding-points from the observations of the leaves is what makes this algorithm suboptimal, while MINC in each partition is optimal.

Algorithm 2 Subtree Decomposition Algorithm:

Consider a tree G , with sources S and receivers R . Each source sends one probe packet. Each receiver receives at most one probe packet.

- Determine the coding points. These partition G into $|T| \leq 2S - 1$ subtrees.
- For each of the $|T|$ subtrees:
 - If the multicast tree is rooted at a coding point:
 - * if any of the descendant receivers receives a probe, use this experiment as a measurement on the subtree.
 - * otherwise, w.p. p assume no node in R received a probe packet, and w.p. $(1 - p)$ ignore the experiment.
 - If the multicast tree is rooted at a source S_i :
 - Consider each coding point \mathcal{C} that acts as a receiver:
 - * if no descendant receivers $\mathcal{C}(R)$ observed a probe, assume, w.p. p , that \mathcal{C} received a packet, and w.p. $(1 - p)$, that it did not.
 - * otherwise
 - if at least one of $\mathcal{C}(R)$ observed a linear combination of x_i , deduce that \mathcal{C} received x_i .

We introduce the probability p in order to account for the fact that if none of the receivers in $\mathcal{C}(R)$ receives a packet, this might be attributed to two distinct events: either the coding point \mathcal{C} itself did not receive a packet, or \mathcal{C} did receive a packet, which got subsequently lost in the descendent edges. For example, in Fig. 26, consider the tree rooted at S_1 ; if R_2 receives x_1 or $x_1 + x_2$, we deduce that x_1 was received at node 4. If R_2 received x_2 , we deduce that x_1 was not received at node 4. If R_2 does not receive a probe packet, then, with probability $1 - p$, we assume that 4 did not receive a probe packet. Ideally, p should match the probability that \mathcal{C} correctly received a probe packet. This depends on the graph structure and on the loss probabilities downstream of \mathcal{C} , and possibly prior information we may have about the link loss rates.

2) **MINC-like Heuristic:** For every multicast node, we can use the MINC algorithm described in [2]. For every coding point, we can use RMINC described in Section V-B.2.

Similarly to the subtree decomposition, we infer which probes have been received by an interior node i from observations at the downstream receivers. In particular, if at least one receiver downstream of i has received a probe with any content (the probe is from at least one source and potentially contains the XOR of probes from multiple sources), then we can infer that i received the packet. This can be used to compute the probability γ_i , in the terminology of MINC [2]. If no downstream receiver got any probe, we decide w.p. p whether the node i received a probe or not, exactly the same as in the subtree decomposition. The reductions shown in Fig. 22 use similar arguments and can serve as examples.

Different from the subtree decomposition, which estimates the α 's locally in each subtree, we use the mapping from γ 's to α 's provided in MINC [2] to estimate the α 's in the entire graph. This heuristic is optimal for multicast and reverse multicast configurations, and for configurations that are concatenations of the two, but suboptimal for any other configuration.

3) **Belief Propagation:** We propose to use a Belief Propagation (BP) approach, similar to what was proposed in [28]. Unlike the previous two heuristics, which are specific to tree topologies, the BP approach also applies to general graphs. The first step in the BP approach is to create the factor graph corresponding to our estimation problem. Fig. 6 shows the factor graph corresponding to the 9-link tree shown in Fig. 26. This is a bipartite graph: on one side there are the links (variable

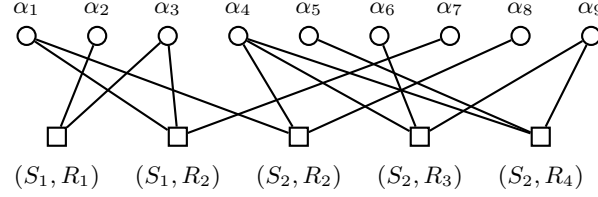


Fig. 6. Bipartite graph corresponding to the 9-link example tree in Fig. 26. It indicates which edges belong to which observable paths.

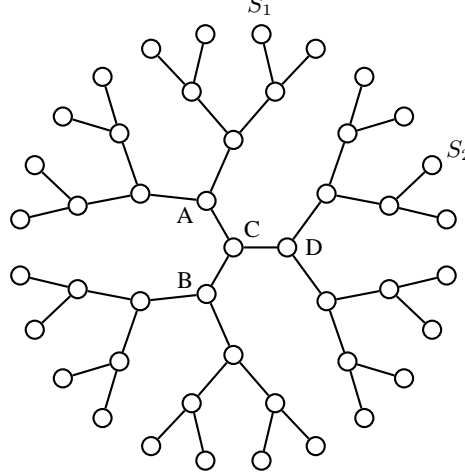


Fig. 7. A tree with 45 links used for simulating the suboptimal estimators.

nodes), whose loss rates we want to estimate; on the other side there are the paths (function nodes) that are observed by each received probe. An edge exists in the factor graph between a link and a path, if the link belongs to this path in the original graph. Note that in tree topologies, there exists exactly one path for every source-receiver pair; while this is not the case in general graphs. Once the factor graph is created from the original graph, each received probe triggers message passing and results in an estimate of link success probabilities; these estimates from different probes are then combined using standard methods [28]. The result is an estimate ($\hat{\alpha}_e$) of the actual success probability (α_e) of every link $e \in E$.

D. Simulation Results

In this section, we evaluate the heuristic estimators via simulation and we compare them to each other as well as to multicast-based tomography. The main finding is that using more than one source helps: using multiple sources and network coding (even with suboptimal estimation) outperforms a single multicast tree (even with optimal estimation), thus demonstrating the usefulness of our approach.⁹

Consider the 45-link topology shown in Fig. 7, where all links have the same success rate α . We will estimate α and compare different methods in terms of their estimation accuracy. First, we did simulations for $\alpha = 0.7$, a large number of probes, and repeated for many experiments. We looked at the mean square error (*MSE*) at each link. The results are shown in Fig. 8 for the following three algorithms:

- a single multicast source S_1 and maximum likelihood estimation (top plot).
- two sources S_1, S_2 , network coding at the middle node C , and the MINC-like heuristic (middle plot).
- the same two sources and coding point, with the subtree estimation algorithm (bottom plot).

Notice that in the case of two sources, the 45-link topology is partitioned into 3 subtrees: one rooted at A (where probe x_1 flows), another rooted at D (where x_2 flows) and a third one rooted at B (where $x_1 + x_2$ flows).

One can make several observations from this graph. First, using two sources and network coding, even with suboptimal estimators, performs better than using a single multicast source and an ML estimator. Indeed the residual entropy (which is the metric that summarizes the *MSE* across all 45 links) is lower for two sources with the MINC-like ($ENT = -317.9$) and for the subtree-decomposition ($ENT = -314.9$) heuristics, than it is for the single source MLE ($ENT = -294.5$). This illustrates the benefit of using multiple sources. Second, notice that the *MSE* for individual links is smaller in the lower two graphs than in the top graph, for all links except for links 43, 44, 45, for which it is significantly higher. This is no coincidence: links 43, 44, 45 are the middle ones (CA, CB, CD in Fig. 7). This is due to the fact that we cannot directly observe the packets received at the coding point C and we have to infer them from observations at the leaves of subtree rooted at B . The

⁹Note that using more than one multicast sources, without network coding, would traditionally require to combine the observations from the two trees in a suboptimal way [3], thus further degrading the performance; that is why we skip the comparison and compare only against a single multicast tree and optimal estimation, which has the best performance among baselines.

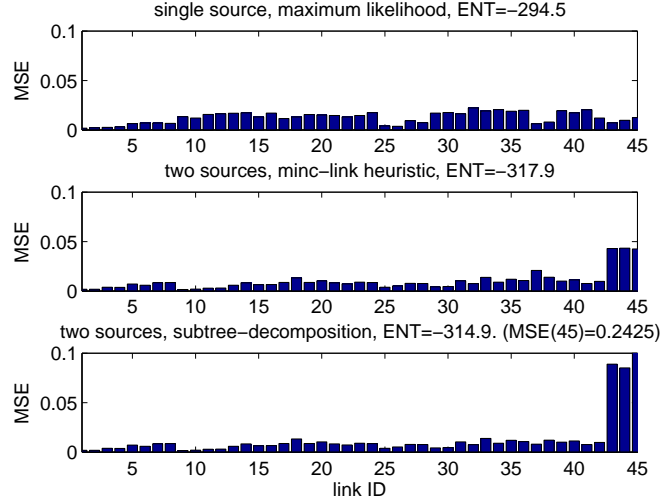


Fig. 8. Comparison of one multicast source + MLE vs. two sources + network coding + suboptimal estimation (subtree decomposition and MINC-like heuristic). We show the MSE for each link in the 45-link topology.

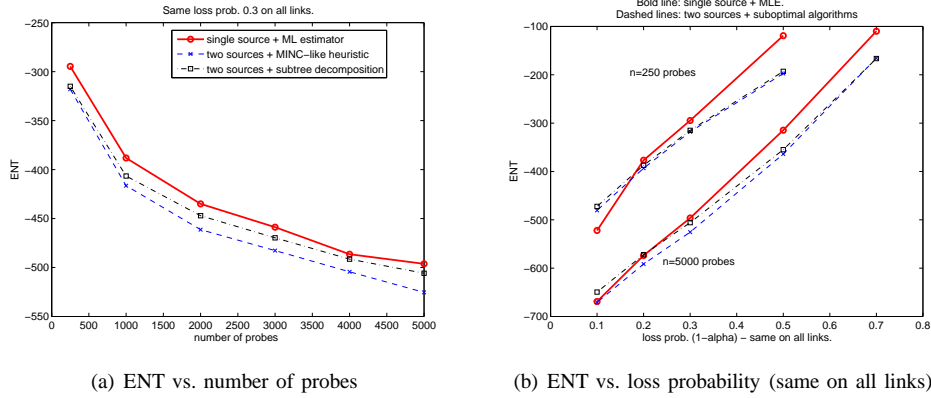


Fig. 9. Comparison of one source with MLE, to two sources with suboptimal estimation (MINC-like and subtree estimation algorithms) for the 45-link tree. The comparison summarizes the error ENT over all links.

performance of the heuristics could further improve by using the following tweak: we could estimate what probes are received at C, using observations from leaves not only in the subtree rooted at B, but also from the subtrees rooted at A and D.

The above simulations were for a single value of $\alpha = 0.7$. We then exhaustively considered several values of α (same on all links) and n (the number of probes). The results are shown in Fig. 9. We can see that, even with suboptimal estimation, using two sources consistently outperforms a single multicast source, even with MLE estimation. This is apparent in Fig. 9, where the ENT metric for the single source (drawn in bold lines) is consistently above the other two algorithms.¹⁰

In Fig. 10, we compare the MINC-like and the BP algorithms over the 45-link network, in terms of the ENT measure, and as a function of the number of probes n . Both algorithms yield better performance (lower ENT values) as the number of sources increases from one to five. The MINC-like algorithm performs better for the multicast tree, in which case it coincides with the ML estimator, as well as for the two source tree. However, belief propagation offers significantly better performance for the case of three and five sources. This trend can be explained by looking at the number of cycles in the factor graph. A cycle is created in the factor graph of a network configuration when (1) two different paths have more than one link in common and (2) a set of m paths, say W_m , covers a set E_m of m links, with each of the paths in W_m containing at least two links in E_m . As the factor graph becomes more and more cyclic, the performance of the sum-product algorithm degrades.

Finally, in Fig. 11, we compare the performance of belief propagation to ML estimation using a single source. We considered two trees: the 45-link and another, randomly generated 200-link tree. Because ENT captures the error over all links, and the two considered topologies have different numbers of links, we use ENT_{av} (defined as the ENT value divided by the number of network links) for a fair comparison of the two topologies. ENT_{av} for the 45 link tree is better (lower) than that of the 200 link tree for a given number of probes. We see that the BP algorithm closely follows the optimal ML estimator, for the

¹⁰Two observations on the ENT metric: First, the differences in the value of ENT are significant, although this is not visually obvious; recall that ENT is defined by taking the sum of the \log of the MSE 's. Second, ENT can be < 0 , it is the differential entropy that matters.

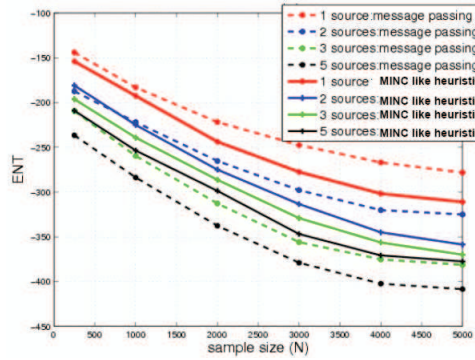


Fig. 10. Estimation error for two suboptimal algorithms (BP and MINC-like) for the 45-link tree. ENT vs. number of probes.

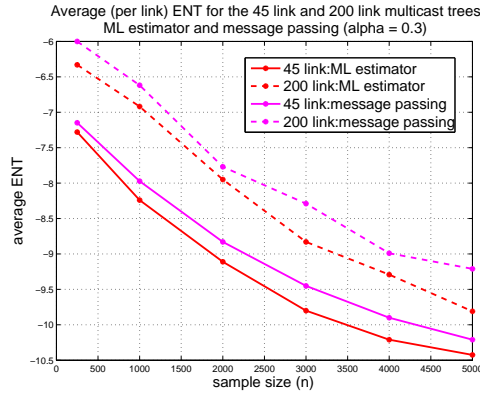


Fig. 11. Comparing BP to MLE for the 45-link and 200-link trees. ENT_{av} is ENT divided by the number of links.

range of number of probes and for both trees considered.

VI. GENERAL TOPOLOGIES

In this section, we extend our approach from trees to general topologies. The difference in the second case is the presence of cycles, which poses two challenges: (i) probes may meet more than once and (ii) probes may be trapped in loops. To deal with these challenges, in this section, we propose (i) an orientation algorithm for undirected graphs and (ii) probe coding schemes, whose design is more involved than in trees.

The approach followed by prior work on tomography over general networks was to cover the graph with several multicast [3] and/or unicast probes [4], [6]. This approach faces several challenges.

- (a) The selection of multicast/unicast probes so as to minimize the total bandwidth (cost) is an NP-complete problem.
- (b) Having several probes from different source-destination paths cross the same link leads to bandwidth waste (especially close to sources or receivers).
- (c) Finding an optimal and/or practical method to combine the observations from different multicast/unicast paths is a non-trivial problem, addressed in a suboptimal way [3].

In contrast, using network coding allows to measure all links with a single probe per link and brings the following benefits:

- (a) It makes the selection of routes so as to minimize cost of linear complexity.
- (b) It eliminates the waste of bandwidth by having each link traversed by exactly one probe per experiment; furthermore, each network coded probe brings more information, as it observes several paths at the same time.
- (c) It does not need to combine observations from different experiments for estimation (as all links in the network are probed exactly once in one pass/experiment).

Because of the aforementioned features, the benefits of the network coding approach compared to traditional tomographic approaches are even more pronounced in general topologies than they were in tree topologies.

In this section, we describe the framework for link loss tomography in general graphs. In particular, we address the four subproblems mentioned in Section III-B: (1) identifiability of links (2) how to select the routing (3) how to perform the code design, and (4) what estimation algorithms to use. We evaluate our approach through extensive simulation on two realistic topologies: a small research network (Abilene), used to illustrate the ideas; and a large commercial ISP topology (Exodus), used to evaluate the performance in large graphs.

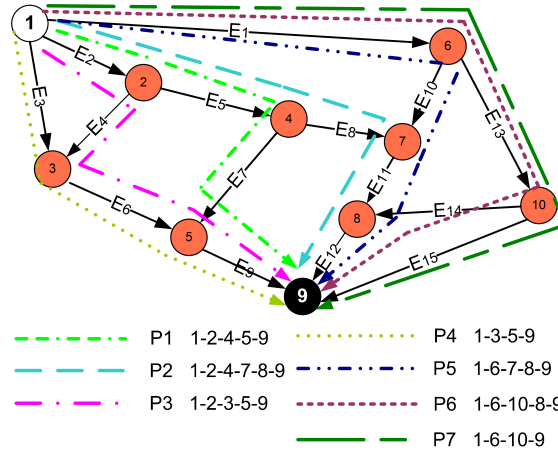


Fig. 12. Example of a general topology (Abilene). For one source (node 1), we show the orientation of edges, the resulting receiver (node 9) and the possible paths from the source to the receiver (P_1, \dots, P_7).

A. Identifiability

The identifiability of an edge given a fixed monitoring scheme (for a known topology, set of sources, receivers, and coding scheme) follows from Theorem 4.1 in Section IV. CD is the edge we would like to identify, and we interpret the edges AC, BC, DE and DF as paths that connect CD to sources and destinations. In particular, we are able to identify the link loss rate of edge CD from the probes collected at the receivers, if we can reconstruct the table associated with one of the cases in Fig. 2 (all tables are provided for completeness in Appendix B.1).

More generally, in a general topology, we can identify the set of paths $\{\mathcal{P}\}$ that connect the sources to all receivers. Let $\mathcal{P}(e)$ denote the set of paths that are routed from a source to a receiver, and employ an edge e . Assume that the receivers can infer which of these paths operated during a given experiment (*i.e.*, non of the links on the path failed) and which did not, by observing the received probes. We call this property *path identifiability*. Note that it is not necessary for edge identifiability that the receivers infer the state of all these paths. However, this is the maximum information that can be extracted from the network to infer link loss rates, and can only increase the reliability of the inferred estimates. Moreover, knowing the state of the paths is particularly well suited for running the belief propagation algorithm that we use for estimation of general graphs: indeed, message-passing in the BP algorithm is triggered by giving the state of the paths as input. Therefore, we will attempt to make the maximum number of path states distinguishable, by appropriate selection of coding. The following example indicates how the selection of a coding scheme can allow more or less path states to be distinguishable at a receiver.

Example 4: Consider the network and edge orientation shown in Fig. 12; this is based on a real backbone topology (Abilene [29]) as it will be discussed in detail in a later section. Node 1 acts as a source and node 9 as a receiver; assume that all intermediate nodes are only allowed to do XOR operations.

Notice that paths P_3 and P_1 overlap twice: on edge E_2 , and later on edge E_9 . If all links in both paths function, the XOR operations “cancel” each other out, resulting in exactly the same observation with both paths being disrupted. More specifically, the following two events become indistinguishable: (i) all edges function: node 5 receives packet x_2 through edge E_7 and packet $x_2 + x_3$ through edge E_6 , and sends packet x_3 through edge E_9 to the receiver; (ii) edges E_4 and E_7 fail, while all other edges function: node 5 only receives packet x_3 from its incoming links, and again sends packet x_3 through edge E_9 to the receiver. On the other hand, if we allow coding operations over a larger alphabet, as in Example 6, these two events result in observing the distinct packets (i) $3x_2 + x_3$ and (ii) x_3 at the receiver. \square

B. Routing

First, we discuss the case where we want to estimate the success rate associated with a specific subset of links, and we express the corresponding optimization problem as a LP that can be solved in polynomial time. Then, we examine the practical special case where we are interested in measuring all links, and which will be the main focus of the rest of the section.

1) **Minimum Cost Routing:** Consider an arbitrary network topology, a given set S of nodes that can act as sources, a given set R of nodes that can act as receivers, and a set I of edges whose link success rate we want to estimate. Our goal is to estimate the success probability for all links in I at the minimum bandwidth cost. That is, we assume that a cost $C(e)$ is associated with each edge e , that is proportional to the flow through the edge. We are interested in identifying the success rate α_e of edge $e \in I$. Let the ρ be the rate of probes crossing that edge, in a manner consistent with the identifiability conditions for edge e .

Remarks. We note that the flow-based formulation of this problem does not rely on any major assumption. The accuracy of estimation depends only on the number of probes and not on the rate of the probe flows. The rates determine how quickly

those n packets will be collected. *E.g.*, for smaller rates, it will take longer to collect the n packets. We also note that having flows coded together in an edge does not reduce the estimation accuracy. In fact, a coded packet observes more than one path, thus increasing the estimation accuracy vs. bandwidth tradeoff.

The minimum cost routing problem was shown to be NP-hard, when performing tomography with multicast trees [30]. Indeed, the problem of even finding a single minimum cost Steiner tree is NP-hard. In contrast, we show here that if we use network coding, we can find the minimum cost routing in polynomial time. In the case of network coding, to ensure identifiability, we want to route flows so that the conditions in Theorem 4.1 are satisfied. We will consider the flow-interpretation of paths in Theorem 4.1, *i.e.*, we will think of each path as a flow of fixed rate ρ . To ensure minimum cost, we want these flows to use the minimum resources possible.

Below we provide a Linear Programming (LP) formulation that allows to solve the minimum cost cover problem in polynomial time, provided that we allow intermediate nodes to combine probes. We assume that there are no capacity constraints on the edges of the network, *i.e.*, we can utilize each edge as much as we want. This is a realistic assumption, since the rate ρ at which we send probe packets would be chosen to be a very small fraction of the network capacity, and nowhere close to consuming the whole capacity.

Intuition. Following an approach similar to [31], we introduce conceptual flows that can share a link without contending for the link capacity. We associate with each edge $e_i \in I$ one such conceptual flow f^i . We would like each f^i to bring probe packets to link $e_i = u_i v_i \in I$ in a manner consistent with the conditions of Theorem 4.1 for edge e_i . We allow conceptual flows corresponding to different edges e_i to share edges of the graph without contention, and will measure through a total flow f the utilization of edges by probe packets. We use the condition $f^i \leq f$ to express the fact that each packet in f might be the linear combination of several packets of conceptual flows.

Notation. Let $C : E \rightarrow R^+$ be our cost function that associates a non-negative cost $C(e)$ with each edge e . We are interested in minimizing the total cost $\sum_e C(e)f(e)$, where $f(e)$ is the flow through edge e . We also denote by $f_{in}(v)/f_{out}(v)$ the total incoming/outgoing flow of vertex v and with $f_{in}(e)/f_{out}(e)$ the total incoming/outgoing flow to edge e . The same notation but with the superscript i , *e.g.*, $f_{in}^i(u)$ has the same meaning but specifically for conceptual flow f^i . We connect all nodes in $S = \{S_i\}$ to a common source node \mathcal{S} through a set of infinite-capacity and zero-cost edges $E_S = \{\mathcal{S}S_i\}$. Similarly, we connect the nodes in $R = \{R_i\}$ to a common node \mathcal{R} using an infinite-capacity and zero-cost set of edges $E_R = \{R_i\mathcal{R}\}$.

We summarize the LP program for Minimum Cost Routing below:

$$\begin{aligned}
& \min \sum_e C(e)f(e) \\
& f(e) \leq \rho \quad \forall e \in E - E_S - E_R \\
& f(e) = \rho \quad \forall e \in I \\
& \text{Each conceptual flow } f^i \text{ corresponding to } e_i = u_i v_i \text{ satisfies the constraints:} \\
& f^i(e) \leq f(e) \quad \forall e \in E - e_i \\
& f^i(e) \geq 0 \quad \forall e \in E \\
& f_{in}^i(\mathcal{S}) = 0 \\
& f_{out}^i(\mathcal{R}) = 0 \\
& f_{in}^i(u) = f_{out}^i(u) \quad \forall u \in V - \{\mathcal{S}, \mathcal{R}, u_i, v_i\} \\
& f_{in}^i(u_i) \geq \rho \quad /* \text{conceptual flow of rate at least } \rho \text{ gets into } (u_i, v_i) */ \\
& f_{in}^i(u_i) + f_{out}^i(u_i) \geq 3\rho \\
& f_{out}^i(v_i) \geq \rho \quad /* \text{conceptual flow of rate at least } \rho \text{ gets out of } (u_i, v_i) */ \\
& f_{in}^i(v_i) + f_{out}^i(v_i) \geq 3\rho
\end{aligned}$$

The idea is to lower-bound the probe rate $f(e)$, in edge e , given the conceptual flows and the condition $f^i(e) \leq f(e)$. Solving this LP will give us a set of flows and paths, for each edge $e = (u_i, v_i)$. To ensure identifiability, we need to additionally select a coding scheme, so that, the flows arriving and leaving at u_i and v_i utilize distinct packets, *i.e.*, from the observable events at the sink, we can reconstruct for edge e the probability of the events of one of cases 1-4 in identifiability.

In summary, the minimum cost routing problem, so as to identify the loss rates of a predefined set of edges I , can be solved in linear complexity when network coding is used, while the same problem is NP-hard without network coding.

2) Routing (including Source Selection and Link Orientation) for Measuring *all* Links : If we are interested in estimating the success rate of *all* identifiable edges of the graph, as opposed to just a restricted set I as in the previous section, we do not need to solve the above LP. We can simply have each source send a probe and each intermediate node forward a combination of its incoming packets to its outgoing edges. This simple scheme utilizes each edge of the graph exactly once per time slot (set of probes sent by the sources) and thus requires the minimum total bandwidth. Moreover, if an edge is identifiable, there exists a coding scheme that allows it to be so. Example 4 and Fig. 12 demonstrate such a situation: the source (node 1) sends one probe per experiment, which gets routed and coded inside the network, crossing each link exactly once, and eventually arriving at the receiver (node 9).

Challenge I: Cycles. One novel challenge we face in general topologies compared to trees is that probes may be trapped in cycles. Indeed, if network nodes simply combine their incoming packets and forward them towards their outgoing links, in a

Algorithm 3 Orientation Algorithm: Given graph $G = (V, E)$ and senders $S \subset V$, find receivers $R \subset V$ and orientation $\forall e \in E$, s.t. there are no cycles and all edges are identifiable.

```

1: for all undirected edges  $e = (s, v_2)$ ,  $s \in S$  do
2:   Set outgoing orientation  $s \rightarrow v_2$ 
3: end for
4:  $R = \{s \in S \text{ that have incoming oriented edges} \}$ 
5:  $V_1 = S$ ;
6:  $V_2 = \{v_2 \in V - V_1 : \text{s.t. } \exists \text{ edge } (v_1, v_2) \text{ from } v_1 \in V_1\}$ 
7: while  $V_2 \neq \emptyset$  do
8:   Identify and exclude receivers: find  $r \in V_2$  without unset edges:  $R := R \cup \{r\}$ ;  $V_2 := V_2 - \{r\}$ 
9:   Find nodes  $U_1 \subset V_2$  that have the smallest number of edges with unset orientation.
10:  Find nodes  $U_2 \subset U_1$  that have the minimum distance from the sources  $S$ . Choose one of them:  $v^* \in U_2$ .
11:  Let  $E^* = \{(v^*, w) \in E \text{ s.t. } w \in V - V_1\}$ 
12:  for all undirected edges  $(v^*, w) \in E^*$  do
13:    set direction to  $v^* \rightarrow w$ 
14:  end for
15:  Update  $V_1 := V_1 \cup \{v^*\}$ 
16:  Update  $V_2 := \{(\text{non-}V_1) \text{ nodes one edge away from current } V_1\}$ 
17: end while

```

distributed manner and without a global view of the network, then probes may get trapped in a positive feedback loop (cycle) that consumes network resources without aiding the estimation process. The following example illustrates such a situation.

Example 5: Consider again the network shown in Fig. 12, but now assume that the orientation of edges E_4 and E_6 were reversed. Thus edges E_4, E_5, E_7 , and E_6 create a cycle between nodes 2, 4, 5, and 3. The probe packets injected by nodes 3 and 2 would not exit this loop. \square

To address this problem, we could potentially equip intermediate nodes with additional functionalities, such as removal of packets that have already visited the same node. This is not practical because it requires keeping state at intermediate nodes; furthermore, such operations would need to be repeated for every set of probes, leading to increased processing and complexity.

We take a different approach: we remove cycles. Starting from an undirected graph $G = (V, E)$, where the degree of each node is either one (leaves) or at least three (intermediate nodes), we impose an orientation on the edges of the graph so as to produce a directed acyclic graph (DAG). Our approach is only possible if we are given some flexibility to choose nodes that can act as sources or receivers of probe packets, among all nodes, or among a set of candidate nodes.

There are many algorithms one can use to produce a DAG. Below we propose our own orientation algorithm, Alg. 3 that, in addition to removing cycles, also achieves some goals related to our problem. In particular, starting from a set of nodes that act as senders $S \subset V$, Alg. 3 selects an orientation of the graph and a set of receivers, so that (i) the resulting graph is acyclic, (ii) a small number of receiver nodes is selected¹¹, which is desired for the efficient data collection and (iii) the resulting DAG leads to a factor graph that works well with belief propagation estimation algorithms. Alg. 3 guarantees identifiability but is heuristic with respect to criteria (ii) and (iii); it is important to note, however, that optimizing for criterion (iii) is an open research problem (as discussed in Section VI-D).

We now describe Alg. 3. We sequentially visit the vertices of the graph, starting from the source, and selecting an orientation for all edges of the visited vertex. This orientation can be thought as imposing a partial order on the vertices of the graph: in a sense, no vertex is visited before all its parent vertices in the final directed graph.

Lines 1 – 3 attempt to set all links attached to the sources as outgoing. If we allow an arbitrary selection of sources, we may fall into cases where sources contain links to other sources. In this case, one of the sources will also need to act as a receiver, i.e., we allow the set S of sources and the set R of receivers to overlap. In the main part of the algorithm nodes are divided into three sets:

- A set of nodes V_1 which we have already visited and have already assigned orientation to all their attached edges. Originally $V_1 := S$.
- A set of nodes V_2 which are one edge away from nodes in V_1 and are the next candidates to be added to V_1 .
- The remaining nodes are either receivers R or just nodes not visited yet $V_3 := V - V_1 - V_2 - R$.

In each step of the algorithm, one node $v^* \in V_2$ is selected, all its edges that do not have an orientation are set to outgoing, and v^* is added to $V_1 := V_1 \cup \{v^*\}$. Notice that the orientation of edges going from V_1 to V_2 is already set. However, a node $v \in V_2$ may have additional unset edges; if it does not have unset edges, then it becomes a receiver $R := R \cup \{v\}$.

We include two heuristic criteria in the choice of $v^* \in V_2$: (i) first we look at nodes with the smallest number of unset edges; (ii) if there are many such nodes, then we look for the node with the shortest distance from the sources S ; if there are still many such nodes, we pick one of them at random. The rationale behind criterion (i) is to avoid creating too many receivers. The rationale behind criterion (ii) is to create a set of paths from sources to receivers with roughly the same path

¹¹Given a set of sources, one can always produce an orientation and a set of receivers that comprise a DAG, which is what Alg. 3 does. Conversely, given a set of receivers one can always produce an orientation and a set of sources that comprise a DAG. If both the sets of sources and receivers are fixed, a DAG may not always exist, depending on the topology.

length. The criteria (i) and (ii) are just optimizations that can affect the estimation performance¹². The algorithm continues until all nodes are assigned to either R or V_1 .

Lemma 6.1: Algorithm 3 produces an acyclic orientation.

Proof: At each step, a node is selected and all its edges which do not have a direction are set as outgoing. This sequence of selected nodes constitutes a topological ordering. At any point of the algorithm, there are directed paths from nodes considered earlier to nodes considered later. A cycle would exist if and only if for some nodes v_i and v_j : v_j is selected at steps $j > i$ and the direction on the undirected edge (v_i, v_j) is set to $v_i \leftarrow v_j$. This is not possible since if there were an edge (v_i, v_j) , it would have been set at the earlier step i at the opposite direction $v_i \rightarrow v_j$. Therefore the resulting directed graph has no cycle. It is possible, however, that there are nodes with no outgoing edges, which become the receivers. ■

We note that the key point that enables us to create an acyclic orientation graph for an undirected graph is that we allow the receivers to be one of the outputs of the algorithm. Notice that a similar algorithm can be formulated for the symmetric problem, where the receivers R are given and the orientation algorithm produces a (reverse) orientation and a set of sources S , s.t. that there are no cycles. However, if both S and R are fixed, there is no orientation algorithm that guarantees the lack of cycles for all graphs.

Lemma 6.2: Algorithm 3 guarantees identifiability of every link in a general undirected graph consisting of logical links (i.e., with degree ≥ 3) and for any choice of sources.

Proof: The proof follows directly from the fact that the degree of each node is greater than or equal to three (assuming logical links only), each edge bringing or removing the same amount of flow. Thus, either the node is a source or a receiver, or the conditions of Theorem 4.1 and Fig. 2 are satisfied. ■

C. Code Design

Challenge II: Code Design affects Identifiability. Another novel challenge that we face in general topologies compared to trees is that simple XOR operations do not guarantee path identifiability, as we saw in Example 4. We deal with this challenge using linear operations over higher field sizes as the following example illustrates.

Example 6: Let us revisit the general topology shown in Fig. 12 and briefly discussed in Example 4. Node 1 acts as a source: for each experiment it sends probes x_1 , x_2 and x_3 through its outgoing edges 1, 2 and 3 respectively. Nodes 2, 4, 6, 10 simply forward their incoming packets to all their outgoing links. Node 3 performs coding operations as follows: if within a predetermined time-window it only receives probe packet x_2 , it simply forwards this packet. The same holds if it only receives probe packet x_3 . If, however, it receives both packets x_2 and x_3 , it linearly combines them to create the packet $x_2 + x_3$ that it then sends through its outgoing edge E_6 . Nodes 5, 7 and 8 follow a similar strategy. If all links are functioning, node 5 sends packet $3x_2 + x_3$, node 7 sends packet $x_1 + x_2$ and finally node 8 sends packet $3x_1 + x_2$. The receiver node 9 observes for each experiment three incoming probe packets. E.g., if it only observes the incoming packet x_3 , it knows that all paths from the source S have failed, apart from path P_4 . Therefore, it infers that no packets were lost on edges E_3 , E_6 , E_9 . □

More generally, we are interested in practical code design schemes that allow for identifiability of all edges in general topologies. We will achieve this goal by designing for path identifiability, which is a different condition. In particular, we are interested in coding schemes that allow us to identify the maximum number of path states. This can be achieved by mapping the failure of each subset of paths to a distinct probe observed at the receivers. For this to be possible (i) the alphabet size must be sufficiently large and (ii) the coding coefficients must be carefully assigned to edges.

Recall that receiver nodes only have incoming edges. Let e_R be an edge adjacent to a receiver R and $\mathcal{P}(e_R)$ be the set of paths that connect all source nodes to receiver R , and have e_R as their last edge. We say that a probe coding scheme allows maximum path identifiability if it allows the receiver R , by observing the received probes from edge e_R at a given experiment, to determine which of the $\mathcal{P}(e_R)$ paths have been functioning during this experiment and which not.

1) *Alphabet Size:* There is a tradeoff between the field size and path identifiability. On one hand, we want a small field size mainly for low computation (to do linear operations at intermediate nodes) and secondarily for bandwidth efficiency (to use a few bits that can fit in a single probe packet). In practice, the latter is not a major problem, because for each probe we can allocate as many bits as the maximum IP packet size, which is quite large in the Internet.¹³ However, for computation purposes, it is still important that we keep the field size as small as possible. On the other hand, a larger field size makes it easier to achieve path identifiability.

For maximum path identifiability, there is the following loose lower bound on the required alphabet size.

Lemma 6.3: Let $G = (V, E)$ be acyclic and let \mathcal{P}_m denote the maximum number of paths sharing an incoming edge of any receiver R , i.e., $\mathcal{P}_m = \max_{e_R} |\mathcal{P}(e_R)|$. Then the alphabet size must be greater than or equal to $\log \mathcal{P}_m$.

Proof: Assume that one of the \mathcal{P}_m paths is functioning while all the others are not. Since two paths cannot overlap in all edges, there exists a set of edge failures such that this event occurs. For the receiver to determine which of the \mathcal{P}_m paths

¹²One could use different criteria to rank the candidates v^* , so as to enforce additional desirable properties. Here we used shortest path from the sources to impose a breath-first progression of the algorithm and paths with roughly the same length. One could also use other criteria to optimize for the alphabet size and/or the complexity and performance of the estimation algorithms.

¹³The MTU (maximum transmission unit) on the Internet is at least 575 Bytes (4800 bits) and up to 1500 bytes (12000bits), including headers. However, in simulation of realistic topologies, we did not need to use more than 18 bits.

function and which ones fail, it needs to receive at least \mathcal{P}_m distinct values. Essentially, the field size should be large enough to allow for distinguishing among all possible paths arriving at each receiver. Therefore, we need a field size $q \geq \mathcal{P}_m$. ■

What the above lemma essentially counts is the number of distinct values that we need to be able to distinguish. This can be achieved using either scalar network coding over a finite field F_q of size q , or using vector linear coding with vectors of appropriate length. *E.g.*, see [24] for an application to the multicast scenario, where scalar network coding over a finite field of size q was treated as equivalent to vector network coding over the space of binary vectors of length $\log q$.

The reader will immediately notice that there is an exponential number of paths and failure patterns. We would like to note that this is not unique to our work, but inherent to tomography problems that try to distinguish between exponentially large number of configurations, *e.g.*, transfer matrices and their failure patterns in the passive tomography [17], [18]. Even in that case, simulations of large topologies, such as Exodus, showed that a moderate field size is sufficient in practice. However, in our case of active tomography, a potentially large alphabet size is needed only if one insists to infer the success rates on *all* links *simultaneously*. In practice, one can infer the success rates on links one-by-one, by carefully selecting the probes and measuring only the corresponding paths, thus creating the “5-link” motivating example, where XOR operations are sufficient.

2) *Code Design*: Having a large alphabet size is necessary but not sufficient to guarantee path identifiability. We also need to assign coefficients $\{c_i\}$ so that the failure of every subset of paths leads to a distinct observable outcome (received probe content). Here we discuss how to select these coefficients.

Consider a particular incoming edge e_R to a receiver R and let n be the number of paths arriving at this edge from source S . Consider one specific path i that connects source S to R via edges $e_{i_1}, e_{i_2}, \dots, e_{i_n}$. The contribution P_i from path i to the observed probe is what we call a *path monomial*, *i.e.*, the product of coefficients on all edges across the path and of probe \mathcal{X}_S sent by source S :

$$P_i = c_{i_1} \cdot c_{i_2} \dots \cdot c_{i_n} \cdot \mathcal{X}_S$$

For simplicity, we use P_i to denote both a path and the corresponding path monomial. Note that, each path consists of a distinct subset of edges; as a result, no path monomial is a factor of any other path monomial. We can collect all the monomials P_i in a column vector $\vec{P}_{e_R} = (P_1, P_2, \dots, P_n)$.

If all paths arriving at edge e_R are working (no link fails), the received probe at that edge is the summation of the contributions $\vec{P} = (P_1, P_2, \dots, P_n)$ from all n paths:

$$\text{Probe received through } e_R \text{ (when no loss)} = P_1 + P_2 + \dots + P_n$$

In practice, however, any subset of these n paths may fail due to loss on some links and the received probe becomes the summation of the subset of paths that did not fail. Let $\vec{X} = (x_1, x_2, \dots, x_n)$ be the vector indicating which paths failed: $x_k = 0$ if path k failed and 1 otherwise. Therefore, the probe received through e_R , in the case of loss, is

$$\text{Probe received through } e_R \text{ (when loss)} = \vec{X} \cdot \vec{P} = \sum_{k=1}^n x_k \cdot P_k,$$

where \vec{X} is the indicator vector corresponding to the loss pattern, *i.e.*, has entry zero if a path fails, and one otherwise. The vector \vec{X} can take 2^n possible values; let \vec{X}_i denote the i^{th} possible value, $i = 0, \dots, 2^n - 1$. To guarantee identifiability, no two subsets, i, j of failed paths should lead to the same observed probe: $\vec{X}_i \cdot \vec{P} \neq \vec{X}_j \cdot \vec{P}$.

Therefore, a successful code design should lead to 2^n distinct probes, one corresponding to a different subset of paths failing. In other words, to guarantee identifiability, the coefficients $\{c_e\}_{e \in E}$ assigned to edges E should be such that: $\vec{X}_i \cdot \vec{P} - \vec{X}_j \cdot \vec{P} \neq 0$, $\forall i, j = 0, \dots, 2^n - 1$. We can write all these constraints together as one:

$$\prod_{i, j=0, \dots, 2^n - 1} (\vec{X}_i \cdot \vec{P}_{e_R} - \vec{X}_j \cdot \vec{P}_{e_R}) \neq 0 \quad (32)$$

Since each $P_i = c_{i_1} \cdot c_{i_2} \dots \cdot c_{i_n} \cdot \mathcal{X}_S$ is a monomial, with variables the coding coefficients $\{c_e\}_{e \in E}$, the left hand side in Eq.(32) is a multivariate polynomial $f(c_1, c_2, \dots, c_{|E|})$ with degree in each variable at most $d \leq 2^n$.

Lemma 6.4: The multivariate polynomial $f(c_1, c_2, \dots, c_{|E|})$ at the left of Eq.(32) is not identically zero.

Proof: The “grand” polynomial is not identically zero because each factor in the product $(\vec{X}_i \cdot \vec{P}_{e_R} - \vec{X}_j \cdot \vec{P}_{e_R})$ is a nonzero polynomial in $\{c_i\}$. Indeed, \vec{X}_i and \vec{X}_j differ in at least one position, say k , corresponding to a monomial P_k . Consider the following assignment for the variables $\{c_i\}$. Assign to all the variables in this monomial a value equal to one. Assign to all other variables $\{c_i\}$ a value of zero. Since no monomial is a factor of any other monomial, this implies that the vector \vec{P}_{e_R} takes value one at position k , and zero everywhere else. Thus, this assignment results in a non-zero evaluation for the polynomial $(\vec{X}_i \cdot \vec{P}_{e_R} - \vec{X}_j \cdot \vec{P}_{e_R})$, and, as a result, this cannot be identically zero. ■

Up to now, we have considered paths that employ the same incoming edge. We can repeat exactly the same procedure for all incoming edges, and generate, for each such edge, a polynomial in the variables $\{c_i\}$. Alternatively, we could also find these polynomials by calculating the transfer matrix between the sources and the specific receiver node using the state-space representation of the network and the algebraic tools developed in [32]. Either way, the code design consists of finding values

Algorithm 4 Deduce State of the Paths from Observations

```

for all  $s \in \text{Senders}$  do
  for all  $r \in \text{Receivers}$  do
    for all incoming links  $e_r$  do
      Map the observed probe to the state of all paths from  $s$  to  $r$  coming through link  $e_r$ 
    end for
  end for
end for
  
```

for the variables $\{c_i\}$ so that the product of all polynomials, f , evaluates to a nonzero value. There are several different ways to find such assignments, extensively studied in the network coding literature, *e.g.*, [33]–[35]. One way to select the coefficients is randomly, and this is the approach we follow in the simulations. In that case, it is well-known that we can make the probability that $f(c_1, c_2, \dots, c_{|E|}) = 0$ arbitrarily small, by selecting the coefficients randomly over a large enough field¹⁴.

Deterministic Operation. We emphasize that although the selection of coefficients may be selected randomly (at setup time), the operation of intermediate nodes (at run-time) is deterministic. At setup time, we select the coefficients and we verify the identifiability conditions, and select new coefficients if needed for the conditions to be met. After the selection is finalized, we learn the coefficients and use the same ones at each time slot. Learning the coefficients is important in order to be able to infer the state of the paths and links.

State Table and Complexity Issues. Once the coefficients are randomly selected, we need to check whether the constraints summarized in Eq.(32) are indeed satisfied. If they are satisfied, the code design guarantees identifiability; if they are not satisfied, then we can make another random selection and check again. One could also start from a small field size and increase it after a number of failed trials.

The evaluation of Eq.(32) above requires to check an exponential number of constraints, up to 2^n , where n is the number of paths for a triplet (source, receiver, edge at receiver). Because the current orientation algorithm does not exclude any edges in the process of building the DAG, we might end up with a large number of paths depending on the connectivity of the topology and the selection of the sources¹⁵. This motivated us to look into more practical approaches¹⁶. Even putting aside the exponential number of paths, for a moment, the problem is essentially a subset sum: we receive a symbol at a receiver and we would like to know which combinations of non-failed paths add up to this number. This is a well-known NP-hard problem.

This being said, we do not expect this to be a source of high complexity in practice for several reasons. First, the algorithm that maps the received symbol to a state of paths can be run offline and the table can be computed and stored. All we need to do every time we receive a symbol is just a table lookup, which is inexpensive ($O(1)$). In other words, we incur setup complexity once in the beginning but not during run time. Second, this design is only necessary if one wants to infer *all* links at the same time, which may be an overkill in practice. The most typical use of our framework in practice will be for inferring the loss rates of a few congested specific links of interest¹⁷. In that case, we do not need to keep track of the state of all paths.

D. Loss Estimation using Belief Propagation

For our approach to be useful in practice, we need to employ a low complexity algorithm that allows to quickly estimate the loss rate on every link from all the observations at the receiver. Because MLE is quite involved for general graphs, especially large ones, we use a suboptimal algorithm instead; in particular, we use the Belief Propagation (BP) approach that we also used for trees, see section V-C.3.

There are two steps involved in the algorithm for each round of received probes. First, from the observations we need to deduce the state of the paths traversed by these probes, as described in Algorithm 4. The second step is to use the Belief-Propagation (BP) algorithm, to approximate Maximum Likelihood (ML) estimation. Once we know which paths work and which failed in this round, we feed this information into the factor graph, which triggers iterations and leads to the estimate of the success rate. Similarly to trees, the factor graph is again a bipartite graph, between links and paths containing these links. For example, Fig. 13 shows the bipartite graph corresponding to the Abilene topology of Fig. 12, which we have been discussing in all the examples in this section.

¹⁴From the Schwartz-Zippel Lemma [33], which has been instrumental for network coding [35], we know the following. If $f(c_1, c_2, \dots, c_{|E|})$ is a non-trivially zero polynomial with degree at most d in each variable, and we choose $\{c_e\}_{e \in E}$ uniformly at random in F_q with $q > d$, then the probability that $f(c_1, c_2, \dots, c_{|E|}) = 0$ is at most $1 - (1 - \frac{d}{q})^{|E|}$.

¹⁵*E.g.*, for the Abilene topology shown in Fig. 12, with 1 source, there were at most three paths per (S, R, e_R) triplet, but for the larger Exodus topology (described in Section VI-E) with 5 sources, the average and maximum number of paths per triplet were 9 and 25 respectively (for a specific selection of sources in both topologies).

¹⁶For example, if we are willing to accept less than 100% path identifiability, we can randomly assign coefficients without checking for identifiability conditions. From the observed probes at the receivers, we then infer the subset of paths that failed by looking up a table which is pre-computed by solving a subset sum problem. If we identify one or more subsets of paths that when failing lead to the same observed probe, we can use a heuristic, *i.e.*, pick one of the candidate subsets, their union or intersection. We then feed the state of the paths to the BP estimation algorithm. This is the approach we follow in the simulation section.

¹⁷This is a well-known fact in the tomography literature, and it is often exploited to set the loss rates of uncongested links to zero and thus reduce the number of unknowns and improve the identifiability.

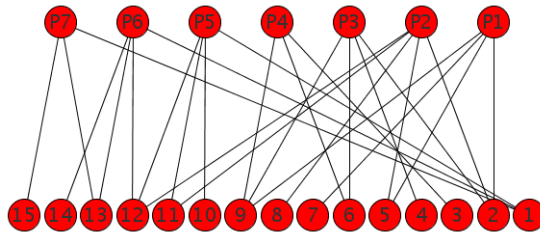


Fig. 13. Factor graph corresponding to the Abilene graph (shown in Fig. 12). It maps the 15 links to the 7 observable paths at the single receiver (9). It is used for the belief propagation estimation algorithm.

The main difference in the general graphs compared to the trees is that there are multiple (instead of exactly one) paths between a source and a receiver; this has two implications. The first implication is that the design of the coding scheme must allow to deduce the state of these multiple paths between a source, receiver and an incoming edge at the receiver (S, R, e_R); this has been extensively discussed in the previous section on code design. The second implication is that there are more cycles in the factor graph of a general graph, which affects the estimation accuracy of the BP algorithm.

In general, the performance of the BP algorithm depends on the properties of the factor graph. Several problems have been identified in the BP literature depending on the existence of cycles, the ratio of factors vs. variables (e.g. links per path) and other structural properties (stopping sets, trapping sets, diameter). Fixing such BP-specific problems are outside the scope of this paper and is a research topic on its own. However, we did address two of the aforementioned problems, using existing proposals from the BP literature. First, for performance enhancement in the presence of cycles in the factor graph, we used a modification of the standard BP, similar to what was proposed in the context of error correcting codes [36]. The idea is to combat the overestimation of beliefs by introducing a multiplicative correction factor $\alpha < 1$ for messages passing between variables (links) and factors (paths)¹⁸. Second, we design the orientation algorithm to traverse the actual topology in a breadth-first manner in order to produce short paths and thus small ratio of links per path in the factor graph, which has a good effect on the BP performance. More generally, we note that the properties of the factor graph depend on the orientation algorithm. One could optimize the orientation algorithm to achieve desired properties of the factor graph. In this paper, we have not done modifications other than the two mentioned above because (i) the overall estimation worked well in all the practical cases we tried and (ii) the design of a factor graph for better BP performance is a research topic on its own and outside the scope of this work.

E. Simulation Results

We now present extensive simulation results over two realistic topologies.

1) *Network Topologies*: We used two realistic topologies for our simulation, namely the backbones of Abilene and Exodus shown in Fig. 14. Abilene is a high-speed research network operating in the US and information about its backbone is available online [29]. Exodus is a large commercial ISP, whose backbone map was inferred by the Rocketfuel project [39]. Both topologies were pre-processed to create logical topologies that have degree at least 3. For Exodus, nodes with degree 2 were merged to create a logical link between the neighbors of these nodes while nodes with degree 1 were filtered; the resulting logical topology contains 48 nodes and 105 links. For the Abilene topology, due to its small size, in addition to some links in tandem merged, more links were added; the modified topology comprises of 10 nodes and 15 links, and is the one shown in Fig. 12 and used as an example of a general topology throughout this section VI.

For all simulations, the link losses on different links are assumed independent, and may take large values as they reflect losses on logical links, comprising of cascades of physical links, as well as events related to congestion control within the network.

2) *Results on the Orientation Algorithm*: In Fig. 15, we consider the Exodus topology and we run the orientation algorithm for all possible placements of one and two sources; we call each placement an “instance”. We are interested in the following properties of the orientation produced by Alg. 3:

- the number of receivers: a small number allows for local collection of probes and easier coordination.
- the number of distinct paths per receiver: this relates to the alphabet size and it is also desired to be small.
- the number of paths per link and links per path: these affect the performance of the belief propagation algorithm.

Fig. 15 shows the above four metrics, sorting the instances first in increasing number of receivers and then in increasing paths/receiver. The following observations can be made. First, the number of receivers produced by our orientation algorithm is

¹⁸In the same way, we could also use an additive correction factor instead. Making those factors adaptive could give even better results. In the same paper [36], additional modifications of the factor graph (junction tree algorithm, and generalized belief propagation) to deal with cycles have been proposed, which we did not implement in this paper. Other possible modifications of the BP include: [37], a multistage iterative decoding algorithm that combines belief propagation with ordered statistic decoding reaches close to the performance of MLE although with a higher complexity than BP; and [38], which uses a probabilistic schedule for message passing between variable nodes and check nodes in the factor graph instead of simple message flooding at every iteration.



Fig. 14. Topologies used in simulation. (a) Left: Abilene Backbone Topology (small research network). (b) Right: Exodus POP Topology (large ISP).

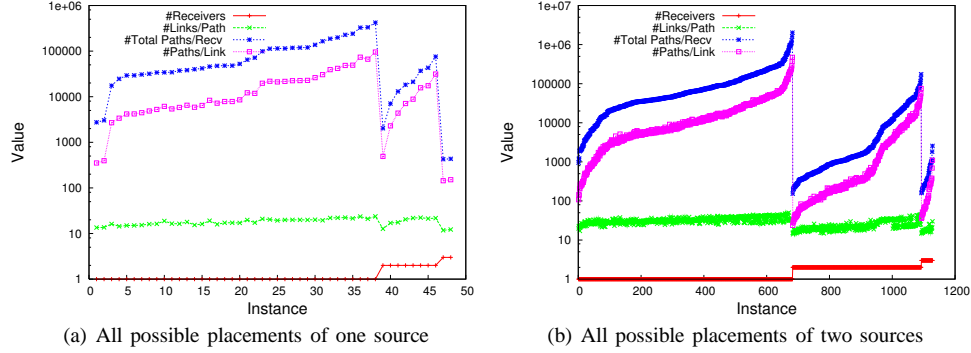


Fig. 15. Running the Orientation Algorithm on the Exodus topology.

indeed very small, as desired. Second, the number of links per path is almost constant, because by construction the orientation algorithm tries to balance the path lengths. Third, the paths/receiver and paths/link metrics, which affect the alphabet size and the quality of the estimation, can be quite high; however, they decrease by orders of magnitude for configurations with a few receivers; therefore, such configurations should be chosen in practice. Finally, Table III considers different choices of sources in the (modified) Abilene and Exodus topologies and shows some properties of the produced orientation.

3) *Evaluation of Random Code Design for Real Topologies:* In this section, we simulate *random code design schemes* for the example topologies of Abilene and Exodus.

Consider a particular incoming edge e_R to a receiver R and let n be the number of paths arriving at this edge from the same source S . If two subsets of paths lead to the same probe, then they are indistinguishable which leads to loss of identifiability. In practice, since many of the paths for a triplet (e_R, S, R) share links between them, we have much less than 2^n possible distinct probes. The exact number depends on the connectivity of the topology. In the simulations, the content of the probe from each subset of paths is used as a key to a hash table. If two subsets lead to the same probe, then they will end up into the same bucket. The number of unique buckets into the hash table gives us the number of different combinations of failed/non-failed paths that are distinguishable from each other. We normalize this number by the total number of possible distinct subsets and

Topology	Srcs-Recvs	Coding Points	Links / Path	Paths / Link	Edge Disj. Paths
Abilene	{1}-{9}	4	3.85	1.8	3
	{5}-{6}	4	3.71	1.73	3
	{9}-{2}	4	4.28	2.0	2
	{1,9}-{7}	5	3.25	1.73	4
	{3,6}-{9}	5	4	2.13	4
	{9,6}-{4}	5	3.25	1.73	4
	{1,5,9}-{7}	5	3.2	2.13	5
	{1,4,10}-{9}	6	3	2.33	6
Exodus	{39,45}-{30,40}	25	9.47	56.47	4

TABLE III

PROPERTIES OF THE ORIENTATION GRAPHS PRODUCED BY ALG. 3 FOR DIFFERENT TOPOLOGIES AND CHOICES OF SOURCES.

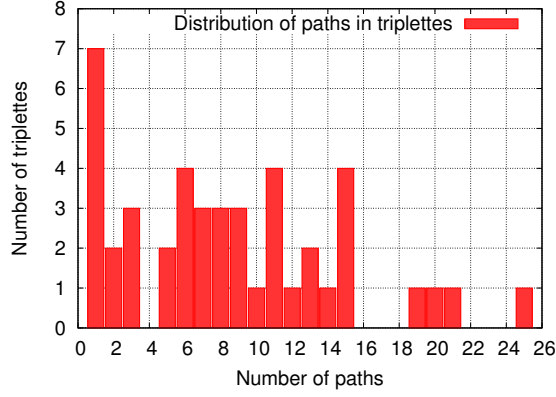


Fig. 16. Distribution of number of paths for all triplets (e_R, S, R) for the Exodus topology

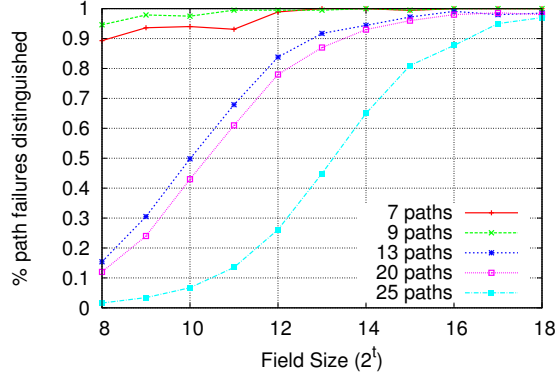


Fig. 17. Random code design for the Exodus topology. The X-axis shows the field size over which we choose the coding coefficients randomly: finite fields with different sizes ($F_{2^8} - F_{2^{18}}$). The Y-axis shows the effect on path identifiability (probability of success, defined as the % of paths in a triplet (S, R, e_R) that we can uniquely distinguish from the observed outcome).

we call this number the probability of success (path identifiability) of the code design for this particular triplet (e_R, S, R) .

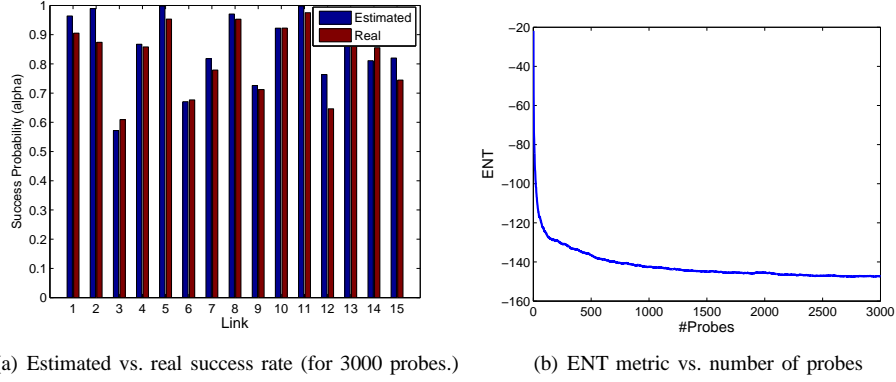
For the *Abilene topology* (10 nodes, 15 links), using one source and the orientation algorithm, we obtained a DAG with one receiver (Fig. 12). The maximum number of paths observed for an incoming edge at the receiver was 3. A random choice of coding coefficients over a finite field of size 2^6 was sufficient to achieve 100% identifiability of all paths on all edges.

For the *Exodus topology* (48 nodes, 15 links), we select 5 sources, apply the orientation algorithm and get three receivers. Fig. 16 shows the distribution of the number of paths for all triplets (e_R, S, R) . There are 16 incoming edges to all three receivers, 44 triplets (e_R, S, R) and 377 paths from the sources to the receivers in total; this leads to 9 paths on average and 25 paths maximum per triplet (e_R, S, R) . We visit all nodes in a random order and we assign coefficients from a finite field with increasing size ($2^{10} - 2^{18}$).

In Fig. 17, we show the probability of success with regards to path identifiability for five such triplets (e_R, S, R) , with 7, 9, 13, 20 and 25 number of paths respectively. The values are averaged over 5 different runs for each field size value. When we use random code selection over a field of size 2^{16} or larger, we get good results: for a field size 2^{18} or higher we get almost 100% success for all triplets. These are good results for a large realistic topology, such as Exodus, since almost 100% success is achieved with much less bits than the 1500bytes of an IP packet. Random assignment of coefficients over a set of prime numbers leads to success probability above 98% when we use up to prime 907 and field size 2^{18} for the linear operations.

4) *Results on Belief-Propagation (BP) Inference:* This section presents results on the quality of the BP estimation for different assignments of loss rates to the links of the two considered topologies.

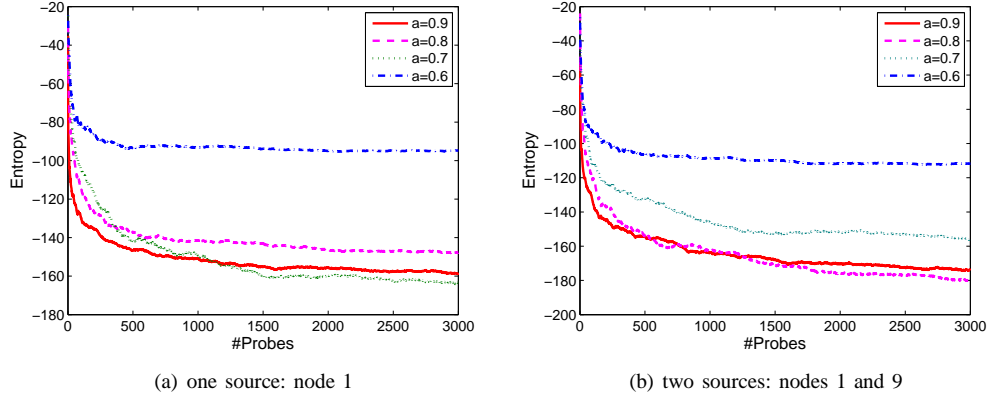
In Fig. 18, we consider the Abilene topology with loss rates inversely proportional to the bandwidth of the actual links; the intuition for this assignment is that links with high bandwidth are less likely to be congested. We see that the estimation error for each link (MSE) and for all links (ENT) decreases quickly. In Fig. 19, the same topology is considered but with the same α on all links: again ENT decreases with the number of probes; as expected, the larger the $\bar{\alpha}$, the slower the convergence; there is not a big difference between having one or two sources in this case. Fig. 20 shows the estimation error ENT for the Exodus topology with uniformly loss rates. Finally, Table IV shows the results for different number and placement of sources in the (modified) Abilene topology. Unlike Fig. 19, Table IV shows that the choice of sources matters and that increasing the number of sources helps in decreasing ENT.



(a) Estimated vs. real success rate (for 3000 probes.)

(b) ENT metric vs. number of probes

Fig. 18. (Modified) Abilene topology. Loss rates ($\bar{\alpha}$'s) are different across links: assigned inversely proportional to the bandwidth of the actual links as reported in [29]. The resulting average loss rate is 17%.



(a) one source: node 1

(b) two sources: nodes 1 and 9

Fig. 19. Abilene topology with the same α on all links.

5) *NC-Tomography vs. Multicast Tomography*: We finally compare the network-coding approach to traditional multicast tomography for general topologies [3]. In the traditional approach, multiple multicast trees are used to cover the general topology, and the estimates from different trees are combined into one, using approaches proposed in [3].

Fig. 21(a) shows the topology we used in the comparison, which is taken from [3]: Nodes $\{0, 1, 2, 5\}$ are sources, nodes $\{12, \dots, 19\}$ are receivers and all remaining nodes (shown as boxes) are intermediate nodes. When the traditional approach is used, probes are sent from each of the four source to all receivers using a multicast tree, an estimate is computed from every tree, and then the four estimates are combined into one using the minimum variance weighted average [3]. When the network coding approach is used, the same four sources and the same receivers are used, but probes are combined at intermediate nodes $\{6, 7\}$. For a fair comparison, the same belief-propagation algorithm has been used for estimation over multicast trees and using the network coding approach. Fig. 21(b) shows the performance of both schemes. We see that the network coding approach achieves a better error vs. number of probes tradeoff. The main benefit in this case comes from the fact that the

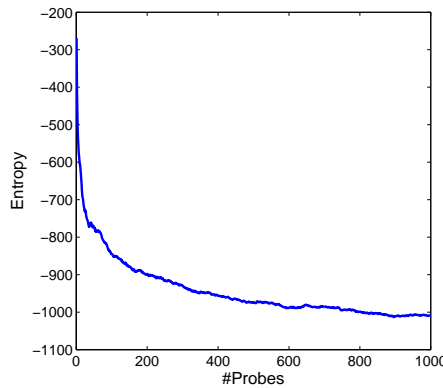


Fig. 20. Exodus topology, considering different loss rates across links: uniformly in $[1\%, 35\%]$.

Srcs-Revs	Entropy for loss rate same over all links					
	$\bar{\alpha} = 0.05$	$\bar{\alpha} = 0.10$	$\bar{\alpha} = 0.15$	$\bar{\alpha} = 0.2$	$\bar{\alpha} = 0.25$	$\bar{\alpha} = 0.30$
$\{1\}-\{9\}$	-178.6	-158.8	-147.9	-147.7	-161.6	-163.5
$\{5\}-\{6\}$	-178.1	-158.3	-149.6	-154.5	-160.4	-156.5
$\{9\}-\{2\}$	-176.1	-163.3	-155.8	-161.2	-166.6	-151.7
$\{1,9\}-\{7\}$	-189.3	-173.9	-166.5	-180.3	-171.7	-156.2
$\{3,6\}-\{9\}$	-186.2	-176.2	-171.3	-177.8	-166.7	-151.4
$\{9,6\}-\{4\}$	-186.9	-174.1	-169.5	-178.7	-173.2	-165.4
$\{1,5,9\}-\{7\}$	-199.8	-190.6	-180.9	-184.4	-172.3	-166.9
$\{1,4,10\}-\{9\}$	-186.4	-183.9	-178.3	-182.3	-177.3	-173.2

TABLE IV

QUALITY OF ESTIMATION FOR THE (MODIFIED) ABILENE TOPOLOGY AND FOR DIFFERENT CHOICES OF SOURCE(S).

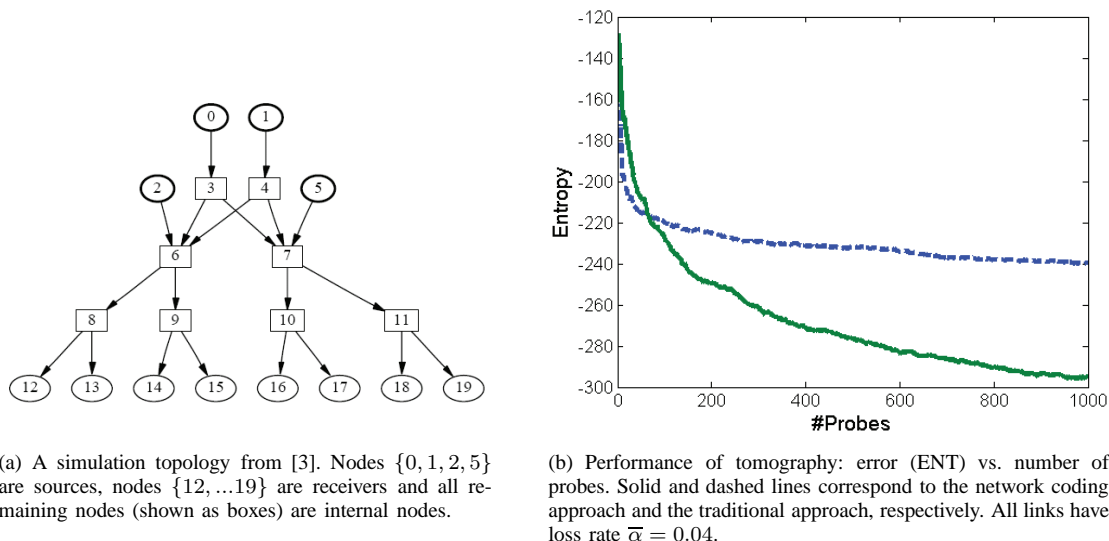


Fig. 21. Comparison of network coding approach to traditional tomography. In both cases, the same sources and receivers are used. In the traditional case, four multicast trees are used and the estimates are combined using methods from [3]. In the network coding case, probes are combined wherever they meet in the network (nodes 6 and 7).

network coding approach eliminates the overlap of the multicast trees below nodes 6 and 7.

There is of course a wealth of other tomographic techniques that are not simulated here. (For example, we could cover a general graph with unicast probes, but this would perform worse than using multicast probes.) The reason is that [3] is directly comparable to our approach and thus highlights the intuitive benefits of network coding, everything else being equal. Network coding ideas could also be developed for and combined with other tomographic approaches.

VII. CONCLUSION

In this paper, we revisited the well-studied and hard problem of link loss tomography using new techniques in networks equipped with network coding capabilities. We developed a novel framework for estimating the loss rates of some or all links in this setting. We considered trees and general topologies. We showed that network coding capabilities can improve virtually all aspects of loss tomography, including identifiability, routing complexity and the tradeoff between estimation accuracy and bandwidth overhead.

ACKNOWLEDGEMENTS

We would like to thank Suhas Diggavi and Ramya Srinivasan for interactions on the problem of source selection.

REFERENCES

- [1] R. Castro, M. Coates, G. Liang, R. Nowak, and B. Yu, "Network tomography: Recent developments," *Statistical Science*, vol. 19, no. 3, pp. 499–517, 2004.
- [2] R. Caceres, N. Duffield, J. Horowitz, and D. Towsley, "Multicast-based inference of network-internal loss characteristics," *IEEE Transactions on Information theory*, vol. 45, no. 7, pp. 2462–2480, 1999.
- [3] T. Bu, N. Duffield, F. Presti, and D. Towsley, "Network tomography on general topologies," *ACM SIGMETRICS Performance Evaluation Review*, vol. 30, no. 1, p. 30, 2002.
- [4] M. Rabbat, R. Nowak, and M. Coates, "Multiple source, multiple destination network tomography," in *Proc. of IEEE INFOCOM*, vol. 3, Hong Kong, HK, 2004, pp. 1628–1639.
- [5] N. Duffield, F. Presti, V. Paxson, and D. Towsley, "Inferring link loss using striped unicast probes," New York, NY, 2001, pp. 915–923.

- [6] M. Coates and R. Nowak, "Network loss inference using unicast end-to-end measurement," in *Proc. of ITC Conference on IP Traffic, Modeling and Management*, Monterey, CA, 2000.
- [7] R. Ahlswede, N. Cai, S. Li, and R. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.
- [8] S. Li, R. Yeung, and N. Cai, "Linear network coding," *IEEE Transactions on Information Theory*, vol. 49, no. 2, pp. 371–381, 2003.
- [9] "The network coding webpage," <http://www.netcod.org/>.
- [10] Y. Vardi, "Network Tomography: Estimating Source-Destination Traffic Intensities from Link Data," *Journal of the American Statistical Association*, vol. 91, no. 433, 1996.
- [11] F. Presti, N. Duffield, J. Horowitz, and D. Towsley, "Multicast-based inference of network-internal delay distributions," *IEEE/ACM Transactions on Networking*, vol. 10, no. 6, pp. 761–775, 2002.
- [12] E. Lawrence, G. Michailidis, and V. Nair, "Statistical inverse problems in active network tomography," *JSTOR*, 2007, vol. 54, pp. 24–44.
- [13] Y. Tsang, M. Coates, and R. Nowak, "Passive network tomography using EM algorithms," in *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, New York, NY, 2001, pp. 1469–1472.
- [14] V. Padmanabhan, L. Qiu, and H. Wang, "Passive network tomography using bayesian inference," in *Proc. of ACM SIGCOMM Workshop on Internet Measurement*, New York, NY, 2002, p. 94.
- [15] C. Fragouli and E. Soljanin, "Monograph on Network Coding: Fundamentals and Applications," in *Foundations and Trends in Networking*. NOW Publishers, 2007, vol. 2.
- [16] T. Ho, B. Leong, Y. Chang, Y. Wen, and R. Koetter, "Network monitoring in multicast networks using network coding," in *Proc. of International Symposium on Information Theory*, 2005, pp. 1977–1981.
- [17] G. Sharma, S. Jaggi, and B. Dey, "Network tomography via network coding," in *Proc. of Information Theory and Applications Workshop*, San Diego, 2008, pp. 151–157.
- [18] H. Yao, S. Jaggi, and M. Chen, "Passive network tomography for erroneous networks: A network coding approach," *arxiv:0908.0711*.
- [19] M. Jafarisiavoshani, C. Fragouli, S. Diggavi, and C. Gkantsidis, "Bottleneck discovery and overlay management in network coded peer-to-peer systems," in *Proc. of SIGCOMM workshop on Internet network management*, Tokyo, Japan, 2007, p. 298.
- [20] C. Fragouli and A. Markopoulou, "A network coding approach to overlay network monitoring," in *Proc. of Allerton Conference*, Monticello, IL, 2005.
- [21] C. Fragouli, A. Markopoulou, R. Srinivasan, and S. Diggavi, "Network Monitoring: it depends on your points of view," in *Proc. of Information Theory and Applications Workshop*, San Diego, CA, 2007.
- [22] M. Gjoka, C. Fragouli, P. Sattari, and A. Markopoulou, "Loss tomography in general topologies with network coding," in *Proc. of IEEE Globecom*, Washington, DC, Nov 2007.
- [23] P. Sattari, A. Markopoulou, and C. Fragouli, "Maximum Likelihood Estimation for Multiple-Source Loss Tomography with Network Coding," in *Proc. of International Symposium on Network Coding (NetCod)*, Beijing, China, July 2011.
- [24] M. Javad and C. Fragouli, "Vector network coding algorithms," in *Proc. of IEEE ISIT*, Austin, TX, June 2010.
- [25] T. Cover and J. Thomas, *Elements of information theory*. Wiley, 2006.
- [26] E. Lehmann, *Elements of large-sample theory*. Springer, 1999.
- [27] J. A. Rice, *Mathematical Statistics and Data Analysis*. Belmont, California: Duxbury Press, 1995.
- [28] Y. Mao, F. Kschischang, B. Li, and S. Pasupathy, "A factor graph approach to link loss monitoring in wireless sensor networks," *IEEE JSAC, Special Issue on Self-Organizing Distributed Collaborative Sensor Networks*, vol. 23, no. 4, pp. 820–829, 2005.
- [29] "The abilene research network," <http://abilene.internet2.edu/>.
- [30] M. Adler, T. Bu, R. Sitaraman, and D. Towsley, "Tree layout for internal network characterizations in multicast networks," 2001, pp. 189–204.
- [31] Z. Li, B. Li, D. Jiang, and L. Lau, "On achieving optimal throughput with network coding," in *Proc. of IEEE INFOCOM*, vol. 3, Miami, FL, 2005.
- [32] R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 782–795, 2003.
- [33] J. Schwartz, "Fast probabilistic algorithms for verification of polynomial identities," *Journal of the ACM*, vol. 27, no. 4, p. 717, 1980.
- [34] N. Harvey, "Deterministic network coding by matrix completion," Master's thesis, MIT, 2005.
- [35] T. Ho, M. Medard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, 2006.
- [36] M. Yazdani, S. Hemati, and A. Banihashemi, "Improving belief propagation on graphs with cycles," *IEEE Communications Letters*, vol. 8, no. 1, pp. 57–59, 2004.
- [37] M. Fossorier, "Iterative reliability-based decoding of low-density parity checkcodes," *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 5, pp. 908–917, 2001.
- [38] Y. Mao and A. Banihashemi, "Decoding low-density parity-check codes with probabilistic schedule," in *IEEE Communication Letters*, 2001.
- [39] "Rocketfuel: an isp mapping engine," <http://www.cs.washington.edu/research/networking/rocketfuel>.

APPENDIX A: PROOFS OF THEOREMS

Appendix A.1: Proof of Theorem 4.1

Proof: To prove that conditions 1 and 2 are necessary, consider that condition 1 is not satisfied. Then C can only receive one stream of probe packets, since it is connected to only one source. There exists an edge e through which this stream of probe packets arrives at node C . The link success rate associated with link CD cannot be distinguished from the link success rate associated with link e . More formally, if α_e is the success probability associated with link e and α_{CD} the success probability associated with link CD , then the variables α_e and α_{CD} appear always together (e.g., in the expression $1 - \alpha_e \alpha_{CD}$ in the probability function P_α). Therefore there are many pairs of values (α_e, α_{CD}) that lead to the same P_α . According to definition 2, this means that link CD is not identifiable. Similar arguments hold for the other conditions and this completes the forward argument.

Next, we prove that conditions 1 and 2 are sufficient for identifying link CD .

First, let us consider Case 1, where Conditions 1(b) and 2(b) are satisfied. The remaining cases are similar and are discussed at the end of this proof. These conditions mean that the paths involving link CD should be as depicted in Fig. 22(a): AC, BC, DE, DF can be either links or paths from/to the sources/receivers respectively. In the latter case (when AC, BC and DE, DF depict paths), the path success probability can be computed from the success rates of the corresponding links. Essentially, Case 1 (also shown in Fig. 2 – 5-links, Case 1) generalizes the motivating example of Section IV, where the links AC, BC, DE, DF are replaced by paths AC, BC, DE, DF with the same success probability.

(a) Real Topology

(b) Multicast Tree

(c) Reverse Multicast Tree

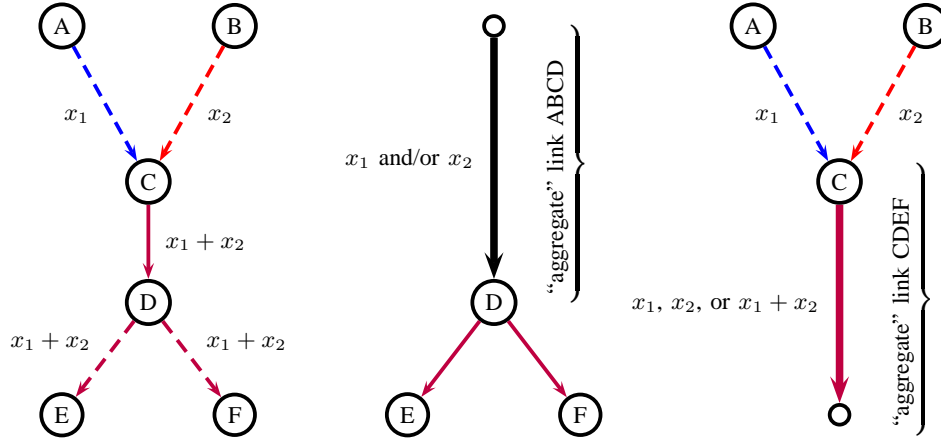


Fig. 22. **Reductions.** (a) depicts the real topology based on conditions 1(b) and 2(b). The goal is to identify the loss rate of link CD . A, B are sources and E, F are receivers. AC, BC, DE, DF can be either links or paths from/to the sources/receivers. In (b), we reduce the real topology to a multicast tree with three links: “aggregate” link $ABCD$ (which transmits *some* symbol, x_1, x_2 or $x_1 \oplus x_2$, below D), and links DE, DF (which broadcast that symbol). In (c), we reduce the real topology to a reverse multicast tree with three links: AC, BC and “aggregate” link $CDEF$ (which transmits the symbol coming in CD to at least one receiver). As shown in detail in Table I, the observations in the reduced topologies are simply unions of disjoint observations in the original topology, and their probabilities are the sum of the probabilities of the corresponding observations in the original topology.

In Definition 2, and consistently with [2], we defined the links as identifiable iff the probability distribution P_α uniquely determines the parameters α^{19} , i.e., iff for $\alpha, \alpha' \in (0, 1]^{|E|}$, $P_\alpha = P_{\alpha'}$ implies $\alpha = \alpha'$. To establish the identifiability of link CD , we repeatedly apply the identifiability result for a 3-link multicast tree (from [2]) and for a reverse multicast tree (leveraging the reversibility property in Theorem 5.1, Section V-B.2). Consider the two reductions of the actual 5-link topology (as described in Section V-B.3), to a multicast tree (MT) shown in Fig. 22(b), and to a reverse multicast tree (RMT) shown in Fig. 22(c), respectively.

In case of the 3-link multicast tree consisting of $ABCD$ and DE, DF , Theorems 2 and 3 in [2] guarantee that α_{DE}, α_{DF} and α_{ABCD} are identifiable. Namely, $P_{\alpha'}^m = P_\alpha^m$ implies $\alpha'^m = \alpha^m$.

On the other hand, since the MLE for the reverse multicast tree has the same functional form as the multicast tree (as described in Section V-B.2), using again the main result of [2], we have that $P_{\alpha'}^r = P_\alpha^r$ implies $\alpha'^r = \alpha^r$.

Proving identifiability in the original topology, via contradiction. Consider the 5-link topology in Fig. 22(a) and assume that there exist $\alpha, \alpha' \in (0, 1]^{|E|}$ for which $P_\alpha = P_{\alpha'}$ and $\alpha \neq \alpha'$.

Use the multicast tree reduction to map success rates α to α^m and associated probabilities P_α to P_α^m . Similarly, reduce the success rates α' to α'^m , and associated probabilities $P_{\alpha'}$ to $P_{\alpha'}^m$. Since $P_\alpha = P_{\alpha'}$, we conclude that $P_\alpha^m = P_{\alpha'}^m$. Because the topology in Fig. 22(b) is identifiable [2], we get that $\alpha^m = \alpha'^m$. This implies that:

$$\alpha'_{DE} = \alpha'^m_{DE} = \alpha^m_{DE} = \alpha_{DE} \quad (33)$$

$$\alpha'_{DF} = \alpha'^m_{DF} = \alpha^m_{DF} = \alpha_{DF} \quad (34)$$

$$(1 - \overline{\alpha'}_{AC} \cdot \overline{\alpha'}_{BC}) \cdot \alpha'_{CD} = \alpha'_{ABCD}{}^m = \alpha^m_{ABCD} = (1 - \overline{\alpha}_{AC} \cdot \overline{\alpha}_{BC}) \cdot \alpha_{CD} \quad (35)$$

Applying similar arguments for the reduction to a reverse multicast tree, we get that $\alpha^r = \alpha'^r$, and as a result:

$$\alpha'_{AC} = \alpha'^r_{AC} = \alpha^r_{AC} = \alpha_{AC} \quad (36)$$

$$\alpha'_{BC} = \alpha'^r_{BC} = \alpha^r_{BC} = \alpha_{BC} \quad (37)$$

$$(1 - \overline{\alpha'}_{DE} \cdot \overline{\alpha'}_{DF}) \cdot \alpha'_{CD} = \alpha'^r_{CDEF} = \alpha^r_{CDEF} = (1 - \overline{\alpha}_{DE} \cdot \overline{\alpha}_{DF}) \cdot \alpha_{CD} \quad (38)$$

From equations (33)-(38), we conclude that $\alpha = \alpha'$, which is a contradiction. Therefore, $P_\alpha = P_{\alpha'}$ implies that $\alpha = \alpha'$, i.e., identifiability.

The remaining cases (combinations of clauses (a), (b), (c) in Conditions 1 and 2, other than 1(b) and 2(b)) are shown in Fig. 2. For example Condition 1(a) or 2(a) correspond to the 3-link multicast or reverse multicast tree and the MINC MLE can then be used directly on these trees. Conditions 1(c) or 2(c) lead to the Cases 2-4 in Fig. 2 and similar reductions as in Case 1 can be used to prove identifiability. This completes the proof. ■

¹⁹Recall that α refers to the vector of all success probabilities, and α_e to the success probability of one particular edge e .

Appendix A.2: Estimating α_{CD}

Proof: Let us denote the outcomes in which link CD has worked by x_{CD} ; the outcomes in which at least one of the upstream paths to C has worked by x_{up} ; and the outcomes in which at least one of the downstream paths after D has worked by x_{dn} . For the intersection of any two of these outcomes, e.g., x_{up} and x_{dn} , we use the notation $x_{up,dn}$. The independence of link loss rates indicates that x_{up} , x_{dn} , and x_{CD} are independent. Therefore:

$$\hat{\alpha}_{CD} = \hat{p}(x_{CD}) = \hat{p}(x_{CD}|x_{up,dn}) = \frac{\hat{p}(x_{CD} \& x_{up,dn})}{\hat{p}(x_{up})\hat{p}(x_{dn})} \quad (39)$$

The numerator equals $1 - \hat{p}([0, \dots, 0]) = \hat{\gamma}_C^r = \hat{\gamma}_D^m$. Also we have that:

$$\begin{aligned} \hat{p}(x_{dn}) &= \hat{p}(x_{dn}|x_{up,CD}) = \hat{p}(x_{dn}|X_D \neq [0, \dots, 0]) \\ &= 1 - \hat{p}(x_{dn}^c|X_D \neq [0, \dots, 0]) = 1 - \prod_{j=1}^Q \bar{\beta}_{d(D)_j}^m \end{aligned} \quad (40)$$

We can derive a similar expression for $\hat{p}(x_{up})$. Therefore:

$$\hat{\alpha}_{CD} = \frac{1 - \hat{p}([0, 0, \dots, 0])}{(1 - \prod_{i=1}^P \bar{\beta}_{f(C)_i}^r)(1 - \prod_{j=1}^Q \bar{\beta}_{d(D)_j}^m)} \quad (41)$$

By writing Eq.(16) for $\bar{\beta}_D^m$ in Fig. 4(a), and by writing Eq.(21) for $\bar{\beta}_C^r$ in Fig. 4(b), we conclude that:

$$1 - \prod_{j=1}^Q \bar{\beta}_{d(D)_j}^m = \frac{\beta_D^m}{\alpha_{agg}^m} = \frac{\gamma_D^m}{A_D^m}, \quad 1 - \prod_{i=1}^P \bar{\beta}_{f(C)_i}^r = \frac{\beta_C^r}{\alpha_{agg}^r} = \frac{\gamma_C^r}{A_C^r} \quad (42)$$

Eq.(27) follows from replacing these results into Eq.(41). ■

Appendix A.3: Proof of Lemma 5.6

Proof: In [2], it is shown that the likelihood function of the reduced multicast tree in Fig. 4(a), $\mathcal{L}^m(\alpha^m)$, can be written as the sum of three distinct parts in which the derivative $\partial \log p^m(x^m)/\partial \alpha_k^m$ is constant. These parts are $\Omega^m(k)$, the $\Omega^m(f^i(k)) \setminus \Omega^m(f^{i-1}(k))$, which we represent by Ω_2^m for simplicity, for $i = 1, 2, \dots, l^m(k)$, and $(\Omega^m(0))^c$. The derivative in these parts is equal to $\frac{1}{\alpha_k^m}$, $\frac{1}{\bar{\beta}_{f^{i-1}(k)}^m} \frac{\partial \bar{\beta}_{f^{i-1}(k)}^m}{\partial \alpha_k^m}$, and $\frac{1}{\bar{\beta}_0^m} \frac{\partial \bar{\beta}_0^m}{\partial \alpha_k^m}$, respectively. Thus, the likelihood equation can be written as:

$$\begin{aligned} \frac{\partial \mathcal{L}^m}{\partial \alpha_k^m} &= \frac{1}{\alpha_k^m} \sum_{x^m \in \Omega^m(k)} n^m(x^m) \\ &+ \sum_{i=1}^{l^m(k)} \left\{ \frac{1}{\bar{\beta}_{f^{i-1}(k)}^m} \frac{\partial \bar{\beta}_{f^{i-1}(k)}^m}{\partial \alpha_k^m} \sum_{x^m \in \Omega_2^m} n^m(x^m) \right\} \\ &+ \frac{1}{\bar{\beta}_0^m} \frac{\partial \bar{\beta}_0^m}{\partial \alpha_k^m} \sum_{x^m \in (\Omega^m(0))^c} n^m(x^m) \end{aligned} \quad (43)$$

Similarly, we can split the likelihood function of the original tree, $\mathcal{L}(\alpha)$, into three parts in which $\partial \log p(x)/\partial \alpha_k$ is constant. These parts will be similar to those of a multicast tree, only with $\Omega^m(k)$ as defined for the original tree in Section V-B.2, and with $l^m(k)$ representing the number of ancestors of node k up to node C (instead of the root 0 in the multicast tree). The derivative $\partial \log p(x)/\partial \alpha_k$ over these parts is also similar to the multicast tree, i.e., $\frac{1}{\alpha_k}$, $\frac{1}{\bar{\beta}_{f^{i-1}(k)}^m} \frac{\partial \bar{\beta}_{f^{i-1}(k)}^m}{\partial \alpha_k}$, and $\frac{1}{\bar{\beta}_C^m} \frac{\partial \bar{\beta}_C^m}{\partial \alpha_k}$, respectively. Thus, we have that:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \alpha_k} &= \frac{1}{\alpha_k} \sum_{x \in \Omega^m(k)} n(x) \\ &+ \sum_{i=1}^{l^m(k)} \left\{ \frac{1}{\bar{\beta}_{f^{i-1}(k)}^m} \frac{\partial \bar{\beta}_{f^{i-1}(k)}^m}{\partial \alpha_k} \sum_{x \in \Omega_2^m} n(x) \right\} \\ &+ \frac{1}{\bar{\beta}_C^m} \frac{\partial \bar{\beta}_C^m}{\partial \alpha_k} \sum_{x \in (\Omega^m(C))^c} n(x) \end{aligned} \quad (44)$$

(i) $\hat{\alpha}_k^m$ vs. $\hat{\alpha}_k$, $k < D$. We first compare the solutions $\hat{\alpha}_k^m$ of Eq.(43) and $\hat{\alpha}_k$ of Eq.(44) for $k < D$. From Eq.(24), we have that:

$$\sum_{x \in \Omega^m(k)} n(x) = \sum_{x^m \in \Omega^m(k)} n^m(x^m) \quad (45)$$

$$\sum_{x \in \Omega_2^m} n(x) = \sum_{x^m \in \Omega_2^m} n^m(x^m) \quad (46)$$

$$\sum_{x \in (\Omega^m(C))^c} n(x) = \sum_{x^m \in (\Omega^m(0))^c} n^m(x^m) \quad (47)$$

Therefore, for any link k located below node D , we have that:

$$\frac{\partial \mathcal{L}^m}{\partial \alpha_k^m} = \frac{\partial \mathcal{L}}{\partial \alpha_k} \implies \hat{\alpha}_k^m = \hat{\alpha}_k, \quad k < D \quad (48)$$

(ii) $\hat{\alpha}_{agg}^m$ vs. $\hat{\alpha}_{CD}$. For α_{agg}^m and α_{CD} , Eq.(43) and Eq.(44) consist of only the first and the last terms. We have:

$$\frac{\partial \mathcal{L}^m}{\partial \alpha_{agg}^m} = \frac{1}{\alpha_{agg}^m} \sum_{x^m \in \Omega^m(D)} n^m(x^m) + \frac{1}{\bar{\beta}_0^m} \frac{\partial \bar{\beta}_0^m}{\partial \alpha_{agg}^m} \sum_{x^m \in (\Omega^m(0))^c} n^m(x^m) \quad (49)$$

$$\frac{\partial \mathcal{L}}{\partial \alpha_{CD}} = \frac{1}{\alpha_{CD}} \sum_{x \in \Omega^m(D)} n(x) + \frac{1}{\bar{\beta}_C^m} \frac{\partial \bar{\beta}_C^m}{\partial \alpha_{CD}} \sum_{x \in (\Omega^m(C))^c} n(x) \quad (50)$$

Thus, $\frac{\partial \mathcal{L}^m}{\partial \alpha_{agg}^m} \neq \frac{\partial \mathcal{L}}{\partial \alpha_{CD}}$, but the definition of $\bar{\beta}_k^m$ indicates that:

$$\bar{\beta}_0^m = 1 - \alpha_{agg}^m (1 - \prod_{j=1}^Q \bar{\beta}_{d(D)_j}^m) \quad (51)$$

$$\bar{\beta}_C^m = 1 - (1 - \prod_{i=1}^P \bar{\beta}_{f(C)_i}^r) \alpha_{CD} (1 - \prod_{j=1}^Q \bar{\beta}_{d(D)_j}^m) \quad (52)$$

From Eq.(45), Eq.(47), Eq.(51), and Eq.(52), we find out that the solutions $\hat{\alpha}_{agg}^m$ of Eq.(49) and $\hat{\alpha}_{CD}$ of Eq.(50) are related via:

$$\hat{\alpha}_{CD} = \frac{\hat{\alpha}_{agg}^m}{1 - \prod_{i=1}^P \bar{\beta}_{f(C)_i}^r} \quad (53)$$

■

Note: Proof of Lemma 5.7 is similar to the proof of Lemma 5.6 above.

Appendix A.4: Proof of Theorem 5.5

Proof: In [2], it is shown that $\hat{\alpha}_k^m$ in Eq.(15) are the MLE of the multicast tree. Therefore, $\hat{\alpha}_k = \hat{\alpha}_k^m$, $k < D$, are also the MLE of the corresponding links in the original tree. In addition, by following the same approach as in [2] and due to the reversibility property, one can show that $\hat{\alpha}_k = \hat{\alpha}_k^r$, $k > C$, are also the MLE of the corresponding links in the original tree. For $\hat{\alpha}_{CD}$, since $\hat{\alpha}_{agg}^m = \hat{A}_D^m$ and using Eq.(42), one can obtain Eq.(27) from Eq.(53). Therefore, Eq.(27) is a solution of $\frac{\partial \mathcal{L}}{\partial \alpha_{CD}} = 0$. Furthermore, from Eq.(50) and Eq.(52), we have that:

$$\frac{\partial^2 \mathcal{L}}{\partial \alpha_{CD}^2} = \frac{-1}{\alpha_{CD}^2} \sum_{x \in \Omega^m(D)} n(x) - \frac{1}{\bar{\beta}_C^{m2}} \left(\frac{\partial \bar{\beta}_C^m}{\partial \alpha_{CD}} \right)^2 \sum_{x \in (\Omega^m(C))^c} n(x) \quad (54)$$

This is always negative. Therefore, \mathcal{L} is concave in α_{CD} and Eq.(27) is the unique solution of the likelihood equation. This solution is also in the desired range $(0, 1]$, because from Eq.(41) we have that:

$$\hat{\alpha}_{CD} > 0 \iff \hat{p}([0, 0, \dots, 0]) < 1$$

i.e., not all packets are lost, which is the default assumption in tomography: no inference can be made without data. In addition:

$$\hat{\alpha}_{CD} < 1 \iff 1 - \hat{p}([0, \dots, 0]) < (1 - \prod_{i=1}^P \bar{\beta}_{f(C)_i}^r) (1 - \prod_{j=1}^Q \bar{\beta}_{d(D)_j}^m)$$

Received at			Is link ok?				
B	E	F	AC	BC	CD	DE	DF
-	-	-	Multiple possible events				
-	-	x	1	0	1	0	1
-	x	-	1	0	1	1	0
-	x	x	1	0	1	1	1
x	-	-	1	1	0	*	*
x	-	-	1	1	1	0	0
x	-	x	1	1	1	0	1
x	x	-	1	1	1	1	0
x	x	x	1	1	1	1	1

TABLE V

CASE 2

Received at		Is link ok?					
B	F	AC	BC	CD	DE	DF	
-	-	Multiple possible events					
-	x_1	1	0	1	0	1	
-	x_2	1	0	0	1	1	
-	x_2	0	*	*	1	1	
-	$x_1 \oplus x_2$	1	0	1	1	1	
x_1	-	1	1	0	0	1	
x_1	-	1	1	*	*	0	
x_1	x_1	1	1	1	0	1	
x_2	x_2	1	1	0	1	1	
x_1	$x_1 \oplus x_2$	1	1	1	1	1	

TABLE VI

CASE 3

This is asymptotically true for $\alpha_{CD} > 0$, because as $n \rightarrow \infty$, the percentage of packets that are *not* lost approaches the probability:

$$(1 - \prod_{i=1}^P \bar{\beta}_{f(C)_i}^r) \alpha_{CD} (1 - \prod_{j=1}^Q \bar{\beta}_{d(D)_j}^m) < (1 - \prod_{i=1}^P \bar{\beta}_{f(C)_i}^r) (1 - \prod_{j=1}^Q \bar{\beta}_{d(D)_j}^m)$$

Therefore, Eq.(27) is the MLE of α_{CD} in the original tree. ■

APPENDIX B: THE EFFECT OF THE NUMBER AND LOCATION OF SOURCES

This appendix provides additional details and simulation results on the effect of the number and location of sources. It extends sections IV-B and V-A.

Appendix B.1: Various configurations for the 5-link topology

Let us consider again the four cases shown in Fig. 2 for the basic 5-link topology. The first case, also shown in Fig. 1, has been discussed in length in Table I and in Section IV. The corresponding tables used for estimation in cases 2, 3 and 4 of Fig. 2 are shown for completeness in Tables V, VI and VII.

Appendix B.2: Simulation Results for the 5-link topology

Consider again the basic 5-link topology of Fig. 2 and focus on estimating the middle link CD. Here we show that, even though with network coding links are identifiable for all four cases, the estimation accuracy differs.

In Fig. 23 we assume that all 5 links have $\bar{\alpha} = 0.3$ and we look at the convergence of the MLE vs. number of probes for *Case 1* (using network coding) and for *Case 2* (multicast probes with source A). Fig. 23(a) shows the estimated value (for

Received at		Is link ok?				
F		AC	BC	CD	DE	DF
-		Multiple possible events				
x_1		1	0	1	0	1
x_2		0	1	1	0	1
x_3		0	0	1	1	1
x_3		*	*	0	1	1
$x_1 \oplus x_2$		1	1	1	0	1
$x_1 \oplus x_3$		1	0	1	1	1
$x_2 \oplus x_3$		0	1	1	1	1
$x_1 \oplus x_2 \oplus x_3$		1	1	1	1	1

TABLE VII

CASE 4

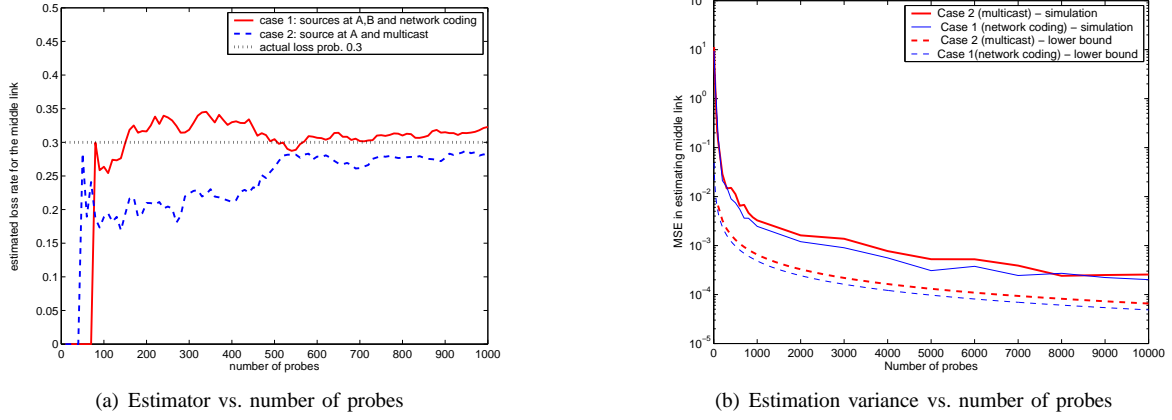


Fig. 23. Convergence of the ML estimator for cases 1, 2

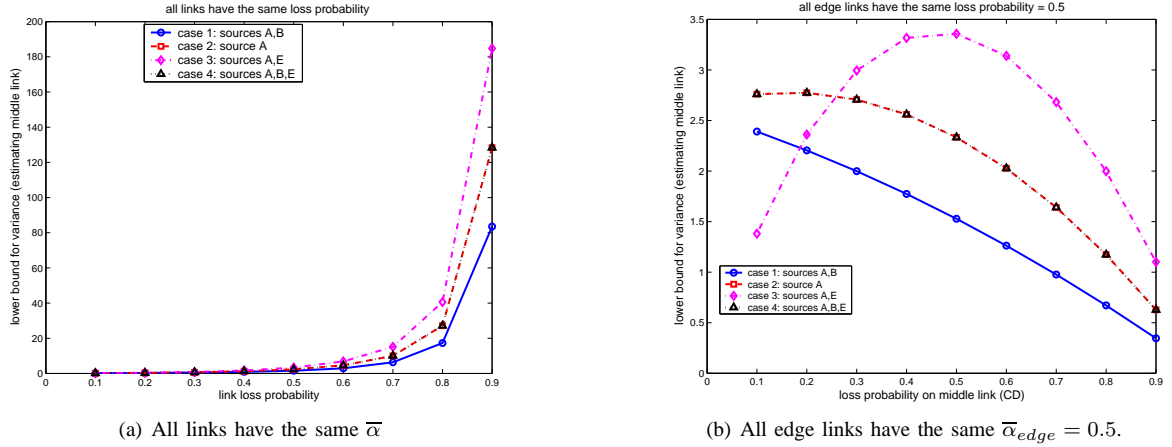


Fig. 24. Comparing the 4 cases in Fig. 2 in terms of the lower-bound of variance.

one loss realization). Both estimators converge to the true value, with the network coding being only slightly faster in this scenario.

In Fig. 23(b) we plot the mean-squared error of the MLE for *Case 1* (using network coding) and for *Case 2* (multicast) across number of probes. For comparison, we have also plotted the Cramer-Rao bound for link *CD*, which is consistent with the simulation results. For this scenario, *Case 1* does slightly better than *Case 2* but not by a significant amount. This motivated us to exhaustively compare all four cases in Fig. 2, for all combinations of loss rates on the 5 links.

Fig. 24 plots the Cramer-Rao bound for the four cases as a function of the link-loss probability on the middle link. The left plot assumes that $\bar{\alpha}$ is the same for five links, while the right plot looks at the case where the edge links have fixed loss rate equal to 0.5. We observe that *Case 1* shows to achieve a lower *MSE* bound. Interestingly, the curves for *Case 2* (multicast) and *Case 4* (reverse multicast) coincide. The difference between the performance of different cases is more evident in the right plot (Fig. 24(b)).

In Fig. 25, we systematically consider possible combinations of loss rates on the 5 links and we show which case estimates better the middle link. In the left figure, we assume that all edge links have the same loss rate and observe that for most combinations of $(\bar{\alpha}_{middle}, \bar{\alpha}_{edge})$, *Case 1* (shown in “+”) performs better. In the right plot, we assume that the middle link is fixed at $\bar{\alpha}_{CD} = 0.8$ and that $\bar{\alpha}_{AC} = \bar{\alpha}_{BC} = \bar{\alpha}_s$, $\bar{\alpha}_{DE} = \bar{\alpha}_{DF} = \bar{\alpha}_r$. Considering all combinations $(\bar{\alpha}_s, \bar{\alpha}_r)$, each one of the four cases dominates for some scenarios. An interesting observation is, again, the symmetry between *Case 2* (multicast) and *Case 4* (reverse multicast).

Appendix B.3: Simulation results for a 9-link example

Example 7: To illustrate these concepts we use the tree shown in Fig. 26. We run simulations for three cases: (1) a multicast tree with the source at node 1 (2) a multicast tree with the source at node 2 (3) two sources at nodes 1 and 2 and a coding point at 4.²⁰ Simulation results are reported for this 9-link topology (and more extensive for a larger 45-link topology) in the simulations subsection V-D. Below we only report the intuition obtained from this exercise. \square

²⁰For the configuration in Fig. 26, the probes could also get combined in node 5 (e.g. depending on link delays or our design). That is, although the choice of sources and receivers automatically determines the orientation of their adjacent links, there may still exist a choice of coding points and orientation for the intermediate links.

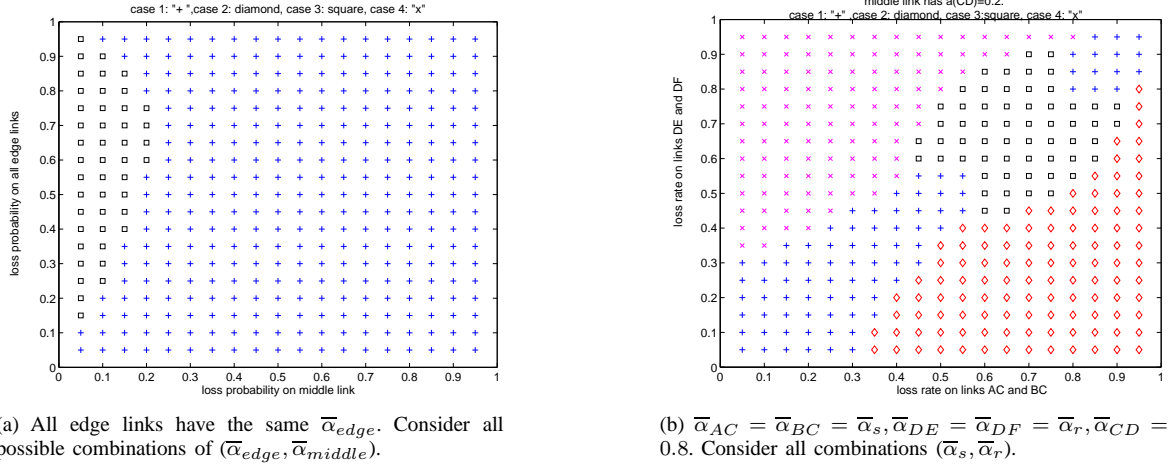


Fig. 25. We indicate which Case (among the four) performs better (has the lowest Cramer-Rao bound), for a given combination of loss rates on all 5 links.

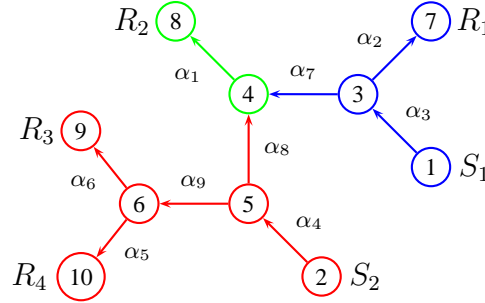


Fig. 26. A network topology with 9 links. The link orientation depicted corresponds to nodes 1 and 2 acting as sources of probes.

First, adding more than one source improves estimation; intuitively, this is because coding points partition the tree into smaller multicast components. Second, the number and placement of sources matter. Third, between two multicast trees with the same number of receivers, better performance is achieved by the tree that is more “balanced” and has the smallest height.

Elaborating on the first observation, note that in trees, each intermediate node is a vertex cut set. For the example of Fig. 26, node 4 decomposes the tree into three components. If node 4 could collect and produce probes, our estimation problem would be reduced in estimating the link-loss rates in three smaller multicast trees: the first tree consisting of source S_1 and receivers R_1 and node 4, the second tree with source S_2 and receiver nodes 4, R_3 and R_4 and the third tree with source node 4 and receiver R_2 . Allowing node 4 to XOR incoming packets approximates this functionality: observing whether R_2 receives a packet that depends on x_1 or x_2 , we can conclude whether node 4 received a packet from S_1 or S_2 respectively.

We made some empirical observations by simulating trees. First, one should select a fraction of sources to receivers that allows to partition the tree into roughly “equal size” subcomponents, where each subcomponent should have at least 2 – 3 receivers. (When links have similar loss rates, “size” refers the number of nodes/links. In general, “size” should also capture how lossy the links in the subcomponent are, resulting in similar loss probabilities for the subcomponents.) Second, one should distribute the sources roughly “evenly” along the periphery of the network.